E-OPPIMISJÄRJESTELMIEN ARKKITEHTUURIMALLIT- SYSTEMAATTINEN KIRJALLISUUSKATSAUS

Ville Murtonen

Tampereen yliopisto
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Ohjaaja: Mikko Ruohonen
Toukokuu 2015

TAMPEREEN YLIOPISTO
Informaatiotieteiden yksikkö
Tietojenkäsittelyoppi
MURTONEN, VILLE: E-oppimisjärjestelmien arkkitehtuurimallit – Systemaattinen kirjallisuuskatsaus
Pro gradu -tutkielma, 55 s.
Toukokuu 2015

Tässä tutkielmassa tarkasteltiin e-oppimisjärjestelmien arkkitehtuurimalleja.

Aihetta alustettiin läpikäymällä erilaisia e-oppimiskäsitteitä, joiden pohjalta muodostettiin näkökulma aiheeseen. Oletus oli, että e-oppimisjärjestelmien arkkitehtuurimalleista löytyisi huomattavan paljon materiaalia. Tämän vuoksi aihetta lähestyttiin systemaattisen kirjallisuuskatsauksen kautta, jotta aineistoa päästiin läpikäymään mahdollisimman kattavasti ja laaja-alaisesti.

Oletus kuitenkin osoittautui tutkimusprosessin myötä virheelliseksi; aiheesta löytyi lopulta valitettavan vähän vertailevaa tutkimusta. Enemmän näyttäytyi käytännön esimerkkejä ja omiin tarpeisiin valittuja ratkaisuja eri toteutuksille.

Lopullinen aineisto muodostui viidestä artikkelista, joiden pohjalta katsaus suoritettiin. Näiden lisäksi valikoitui artikkeleiden pohjalta mukaan vielä yksi lähde tukemaan löydöksiä. Aineiston pohjalta tehtyjen löydösten ja havaintojen perusteella pyritään vastaamaan tutkimuskysymykseen: Millaisia malleja e-oppimisarkkitehtuuriin löytyy kirjallisuuden perusteella?

Aineistosta selvisi myös, että vaikka suoria vertailuja ei ollut tehty, eri kirjoittajien malleista löytyi kuitenkin paljon yhteneväisyyksiä, eritoten osien uudelleenkäytettävyyden ja modulaarisuuden osalta. Löydösten ja havaintojen lisäksi löytyi useita aiheita, joiden pohjalta on mahdollista tehdä jatkotutkimusta esimerkiksi siitä, millaisia ominaisuuksia tulisi löytyä uusista e-oppimisarkkitehtuurimalleista.

Avainsanat: e-learning, e-oppiminen, arkkitehtuuri, arkkitehtuurimalli, Arlene Fink, Okolin, Schabram

Sisällysluettelo

1. Johdanto

Tämän pro gradu -työn aihe syntyi ikään kuin vahingossa; tarkoituksena oli alun perin tutkia e-oppimisjärjestelmiä osana suurempaa kokonaisuutta, mutta mitä syvemmälle kirjallisuusanalyysiin eksyin, sitä enemmän aineistoa alkoi olla kasassa. Jossain vaiheessa kirjallisuusselvityksen aineistoa olikin jo huomattavasti enemmän kuin tutkimuksessa, jota sen oli tarkoitus täydentää. Jo alustavat löydökset osoittivat, että pelkästään tätä aihetta tutkimalla ja analyysia hieman laajentamalla voisi päätyä merkittäviin tuloksiin.

Suurimpana syynä aineiston suureen määrään ja alustaviin löydöksiin oli käytettyjen mallien järjestelmällisyys; kumpaakaan mallia ei voinut seurata ”kevyesti” tai jättää kohtia huomioimatta, vaan käytännössä mallit ohjasivat tekemään metodologisesti tiettyjä asioita, joiden seurauksena työ ikään kuin kirjoitti itsensä. Ainoastaan aineiston valinnan sekä löydösten kuvaamisen kannalta mallit jättivät valinnanvapautta, vaikka näidenkin vaiheiden määrittelyt olivat melko kattavat.

Kaikkiaan mallien avulla työn kirjoittaminen oli sujuvaa ja yksiselitteistä; ne eivät jättäneet liikaa tulkinnanvaraa, mutta tarjosivat silti sen verran liikkumavaraa, että työn pystyi toteuttamaan tässä laajuudessa. Tarkkaan mallia seuraamalla tutkimusta olisi voinut jatkaa vuosikausia ja ottaa siihen mukaan uusia henkilöitä, mutta tähän ei ollut aikaa eikä se olisi varsinaisesti palvellut tarkoitustakaan.

Työn otsikko ja sisältö tarkentuivat lopulta varsinaisen aineiston ja löydösten pohjalta. Alun perin tarkoituksena oli tehdä enemmän vertailevaa tutkimusta arkkitehtuureiden välillä, mutta aineisto ohjasi enemmän yleistä arkkitehtuurimallien katsausta kohti. Näin ollen työn tarkoitukseksi muodostuikin systemaattisen kirjallisuuskatsauksen toteuttaminen erilaisiin e-oppimisjärjestelmien arkkitehtuurimalleihin. Työn tarkoituksena on siis selvittää, millaisia malleja e-oppimisarkkitehtuuriin löytyy kirjallisuuden perusteella, ja miten aihetta on ylipäätään käsitelty kirjallisuudessa. Lisäksi sen tarkoituksena on tuoda esille löydöksiä aineistosta, päätelmiä näiden löydöksien pohjalta sekä jatkotutkimuksen aiheita.

Työstä on pyritty rajaamaan pois kaikki aineisto ja materiaali, joka käsitteli varsinaisia toteutusteknologioita tai valmiita tuotteita, sillä näiden katsottiin olevan alkuperäisen rajauksen ulkopuolella.

Metodina työn toteuttamiseen käytin Finkin [2005] kahdeksan kohdan systemaattista kirjallisuuskatsausta täydennettynä Okolinin ja Schabramin [2010] mallilla. Finkin malli on yleisesti käytetty systemaattisen kirjallisuuskatsauksen malli, ja täydennettyä Okolinin ja Schabramin mallilla se tuntui soveltuvan erinomaisesti juuri oppimista käsittelevän katsauksen suorittamiseen.

Työn rakenteessa on tarkoitus ensiksi tutustuttaa lukija keskeisiin e-oppimisen käsitteisiin. Näin syntyy ymmärrys siitä, mistä näkökulmasta aihetta lähestytään, koska käsitteitä voidaan tulkita tietyissä tilanteissa eri tavoin. Käsitteiden jälkeen läpikäydään tutkimusmenetelmä ja avataan sen eri vaiheet sekä kuvataan, mitkä niistä toteutuivat tässä tutkimuksessa. Käytännössä ajatuksena on, että vaiheiden kuvausten myötä kuka tahansa pystyisi toteuttamaan tutkimuksen uudelleen ja saisi samat tulokset. Ainoana rajoituksena tähän on tietokannoista tehdyt haut, joiden tulokset muuttuvat siinä määrin, että täysin samoja lähtöaineistoja on mahdotonta saada kun hakuja toistetaan myöhemmin. Kuten kappaleessa 4.3. kuitenkin selvitetään, tällä ei välttämättä ole merkitystä tutkimuksen toistettavuuden kannalta.

Tutkimusmenetelmän kuvaamisen jälkeen avataan lukijalle tutkimustulokset, jotka on jaoteltu löydösten mukaan. Ajatuksena tässä on, että lukija pystyisi jo tässä vaiheessa hahmottelemaan yhtenäisyyksiä aineistojen sisällä, ja sitä kautta pystyisi paremmin ymmärtämään yhteenvetoa ja löydöksiä. Jokaisessa kappaleessa on käyty läpi kappaleen aiheeseen parhaiten sopivien aineistojen sisältö ja pyritty avaamaan lukijalle niiden ydinkohdat ja keskinäiset yhtäläisyydet. Tutkimustulosten päätteeksi avataan vielä lyhyesti yleisiä teknologiavalintoja, mutta nämä avataan vain yleisellä käsitetasolla, koska varsinaiset teknologiavalinnat olivat työn rajauksen ulkopuolella.

Yhteenvedossa katsotaan työtä retrospektissä. Tarkoituksena on tuoda lukijalle ymmärrys työn aineiston laadusta, alkuperäisen tutkimuskysymyksen valinnan onnistumisesta, tutkimuksen tuloksista ja löydöksistä ja lopuksi vielä esittää muutamia mahdollisia jatkotutkimuksen aiheita. Yhteenvedon yhtenä tarkoituksena on myös esittää lukijalle kooste koko työstä ja sitä kautta auttaa mahdollisia jatkotutkimuksen tekijöitä pääsemään nopeasti kärryille ja uuden tutkimuksen alkuun.

2. Keskeiset käsitteet

Tutkielma keskittyy e-oppimisjärjestelmien arkkitehtuurimalleihin tietojärjestelmäkontekstissa ja pyrkii selvittämään millaisia malleja on käsitelty kirjallisuudessa vuosina 2004–2014. Tässä kappaleessa määritellään tutkielman kannalta oleelliset käsitteet sellaisina kuin ne soveltuvat tähän kontekstiin. Useita keskeisiä käsitteitä on mahdollista käyttää myös muissa konteksteissa, joten tässä käytetyt määritelmät sekä niiden lähteet on valittu siksi, että ne tukevat e-oppimista ja arkkitehtuuria juuri tietojärjestelmäkonteksissa.

2.1. E-oppiminen (E-learning)

E-oppimisen käsite on hyvin monipuolinen ja käsittää eri asioita eri konteksteissa. Tämän työn osalta e-oppiminen on pyritty rajaamaan nimenomaan verkossa tapahtuvaan oppimiseen ja siihen liittyviin seikkoihin. Selkein määritelmä e-oppimiselle on ehkä Kallialan [2002] kuvaus:

> *”E-oppiminen, e-opetus, eLearning, verkko-oppiminen tai verkko-opetus on oppimista ja opetusta, tiedon hakemista, soveltamista ja ymmärtämistä verkon (käytännössä Internetin) avulla.”*

Kallialan [2002] määritelmä on hyvin yksinkertainen, mutta kuitenkin tarpeeksi kattava, jotta se soveltuu käytettäväksi tämän työn kontekstissa. Erityisesti tämän määritelmän tuoma näkökulma siihen, että e-oppimisessa on kyse myös tiedon soveltamisesta ja ymmärtämisestä, on oleellinen, etenkin kun mietitään arkkitehtuurimalleja käyttäjänäkökulmasta.

E-oppimisen osana on myös verkko-opetus, jonka Kalliala [2002] määrittelee seuraavasti:

> *”Verkko-opetus on tieto- ja viestintätekniikan, erityisesti verkkotekniikan, hyödyntämistä oppimisessa ja opetuksessa. Monimuoto-opetus ja etäopetus sen osana ovat tärkeitä verkko-opetuksen ja -oppimisen muotoja.”*

Verkkotekniikan näkökulman mukaan tuominen on oleellista, koska sen pohjalta määrittyvät myös monet e-oppimisjärjestelmien vaatimuksista. Näitä vaatimuksia vastaan taas joudutaan miettimään ratkaisuja eri arkkitehtuurimalleissa.

Rouse [2005] määrittelee artikkelissaan seuraavasti:

> *”Verkko-pohjainen oppiminen (e-oppiminen), on missä tahansa, milloin tahansa, tapahtuvaa ohjeiden välittämistä internetin, tai yrityksen intranetin välityksellä käyttäjien selaimiin. Oppiminen jakaantuu kahteen malliin; synkroniseen (ohjaaja fasilitoi), ja asynkroniseen (itseopiskelu, itseohjautuva). Ohjeita välitetään yhdistelmällä erilaisia staattisia välineitä (oppimisportaalit, linkitetyt sivustot, video-ohjeet, streamit ja muut vidoepalvelut) ja interaktiivisia välineitä (reaaliaikaiset keskustelut, forumit, video-konferenssit).”*

Rouse [2005] laajentaa vielä entisestään e-oppimisen määritelmää ja käytännössä määrittelee sen tasolle, jossa melkein mikä tahansa organisaation tiedonvälitys voidaan nähdä jollakin tapaa e-oppimisena. Lisäksi Rouse [2005] ottaa kantaa eri oppimismalleihin ja suoranaisiin välineisiinkin. Näiden avulla voidaan tuoda näkökulmaa erilaisten organisaatioiden tarpeisiin ja laajentaa e-oppimista poispäin pelkästä koulumaailman välineestä myös muiden organisaatioiden käyttöön.

Käytännössä voidaan kuitenkin sanoa, että e-oppiminen käsittää kaikki ne tekijät, jotka mahdollistavat henkilön opiskelun ja tiedon saavuttamisen ja välittämisen jossain muussa kuin perinteisessä fyysisessä luokkamallissa.

2.2. Arkkitehtuuri

Arkkitehtuuri on toinen tutkimuksen kahdesta pääkäsitteestä, minkä takia on tärkeätä ymmärtää, mitä arkkitehtuurilla tarkoitetaan tässä kontekstissa. Perinteisestihän arkkitehtuuri nähdään suunnitelmana tai piirrustuksena, jonka pohjalta jokin toteutus tehdään. Koska e-oppimisjärjestelmät ovat hyvin pitkälle ohjelmistoja, käytän arkkitehtuurin määritelmän pohjana tietojärjestelmäarkkitehtuurin määritelmää:

> *”Tietojärjestelmän arkkitehtuuri kuvaa kohdealueensa rakenneosat, niiden ulospäin näkyvät ominaisuudet ja niiden väliset yhteydet ja riippuvuudet. Arkkitehtuuri muodostaa rungon järjestelmän suunnittelulle ja toteutukselle sekä ohjaa järjestelmän rakenteen kehittämistä järjestelmän elinkaaren ajan. Se toimii myös keskusteluvälineenä järjestelmän kehittämisen ja ylläpitämisen sidosryhmien (organisaation johto, käyttäjät, suunnittelijat, toteuttajat) välillä.”* [Wikipedia, 2015.]

Määritelmä on hieman vaikeaselkoinen, mutta se voidaan paloitella hieman selkeämmäksi. Yhtenä osana arkkitehtuuria ovat siis järjestelmän ominaisuudet; ne voidaan nähdä yksittäisinä saarekkeina, joilla on tietty määrä yhteisiä tai yksityisiä ominaisuuksia ja erinäisiä yhteyksiä ulospäin, ja näiden yhteyksien kautta saarekkeet kommunikoivat toisten saarekkeiden kanssa. Saarekkeita voi toteutuksessa olla useita, ja niiden väliset yhteydet ja riippuvuudet muodostavat lopulta koko järjestelmän. Tämä saarekkeisiin perustuva osa määritelmää on hyvin yleispätevä ja sen voidaankin nähdä toteutuvan lähes jokaisessa toteutuksessa käytettävästä arkkitehtuurin määritelmästä riippumatta.

Toisena osana arkkitehtuurin määritelmää on arkkitehtuurin rooli toteutuksen pohjana; se toimii eräänlaisena ohjenuorana ja säännöstönä, jota seuraten ja johon nojaten toteutuksia voidaan rakentaa. Arkkitehtuuri pitää huolen, että jokainen toteutus on toteutettu tiettyjä sääntöjä mielessä pitäen, ja se tarjoaa pohjan kommunikoida toteutuksen tekijöiden kesken järjestelmän ominaisuuksista.

Kansainvälisiä standardeja määrittelevän ANSI/IEEE:n [2000] standardin mukaan arkkitehtuuri määritellään seuraavasti:

> *“Arkkitehtuuri on määritelmänä se suositeltujen käytäntöjen joukko, joka on yrityksen organisaation ytimessä, sen komponenttien sisällä, ja joka kuvaa niiden suhteita toisiinsa, ja jossa on mukana ne periaatteet joilla ympäristöä suunnitellaan, ja kehitetään.”*

Tämä määritelmä on ehkä vielä vaikeatulkintaisempi kuin aiemmin esitellyt. Käytännössä se voidaan kuitenkin tulkita samalla tavalla kuin Wikipedian [2015] määritelmä; arkkitehtuuri on säännöstö, joka kuvaa järjestelmän ominaisuudet ja niiden säännöt ja rajoitukset. Jokaisen toteutuksen, joka pohjaa

määriteltyyn arkkitehtuuriin, pitäisi toteuttaa tai vähintään ottaa huomioon kyseisen arkkitehtuurin säännöt ja määritykset.

2.2.1. Arkkitehtuurimalli

Arkkitehtuurimallilla tarkoitetaan määriteltyä mallia, jonka pohjalta/säännöillä jotain asiaa (tässä tapauksessa e-oppimista) kuvataan. Esimerkiksi ”Learning Architecture Standard Model”. Arkkitehtuurimalli määrittelee usein jonkin kokonaisuuden arkkitehtuurin osat, käytetyt tasot, komponentit ja mahdollisesti teknologian. Se on siis laajempi kokonaisuus kuin pelkkä yksittäinen arkkitehtuuri ja ottaa usein kantaa myös sellaisiin asioihin, joita arkkitehtuurilla ei normaalisti tarkoiteta.

Arkkitehtuurimalli voidaan nähdä kokoelmana sääntöjä, ohjeita, suosituksia ja määritelmiä, joiden pohjalta yksittäistä arkkitehtuuria voidaan lähteä kuvaamaan ja toteuttamaan. Arkkitehtuurimallin tarkoitus on helpottaa ja yhtenäistää käytäntöjä, joiden pohjalta asioita lähestytään ja joiden mukaisesti asioita kuvataan. Arkkitehtuurimalli voi esimerkiksi määritellä vain käytettävän menetelmän, tai se voi mennä syvälle varsinaiseen toteutukseen ja määrittää esimerkiksi käytettävän ohjelmointikielen. Syvemmälle menevien arkkitehtuurimallien päälle on usein helppoa ja nopeaa rakentaa tarkkoja arkkitehtuureja, koska ne ovat vieneet määrittelyn hyvin lähelle varsinaista toteutusta.

Malik [2006] määrittelee artikkelissaan arkkitehtuurimallin seuraavasti:

> *”Arkkitehtuurimalli on rikas ja tarkka diagrammi, joka on luotu käyttäen olemassa olevia standardeja, ja jossa pääasiallinen tarkoitus on kuvata vaihtoehtoja järjestelmän rakenteessa ja mallissa tai ekosysteemissä. Arkkitehtuurisia malleja voidaan käyttää kommunikointiin muiden kanssa, ja palautteen saamiseen.”*

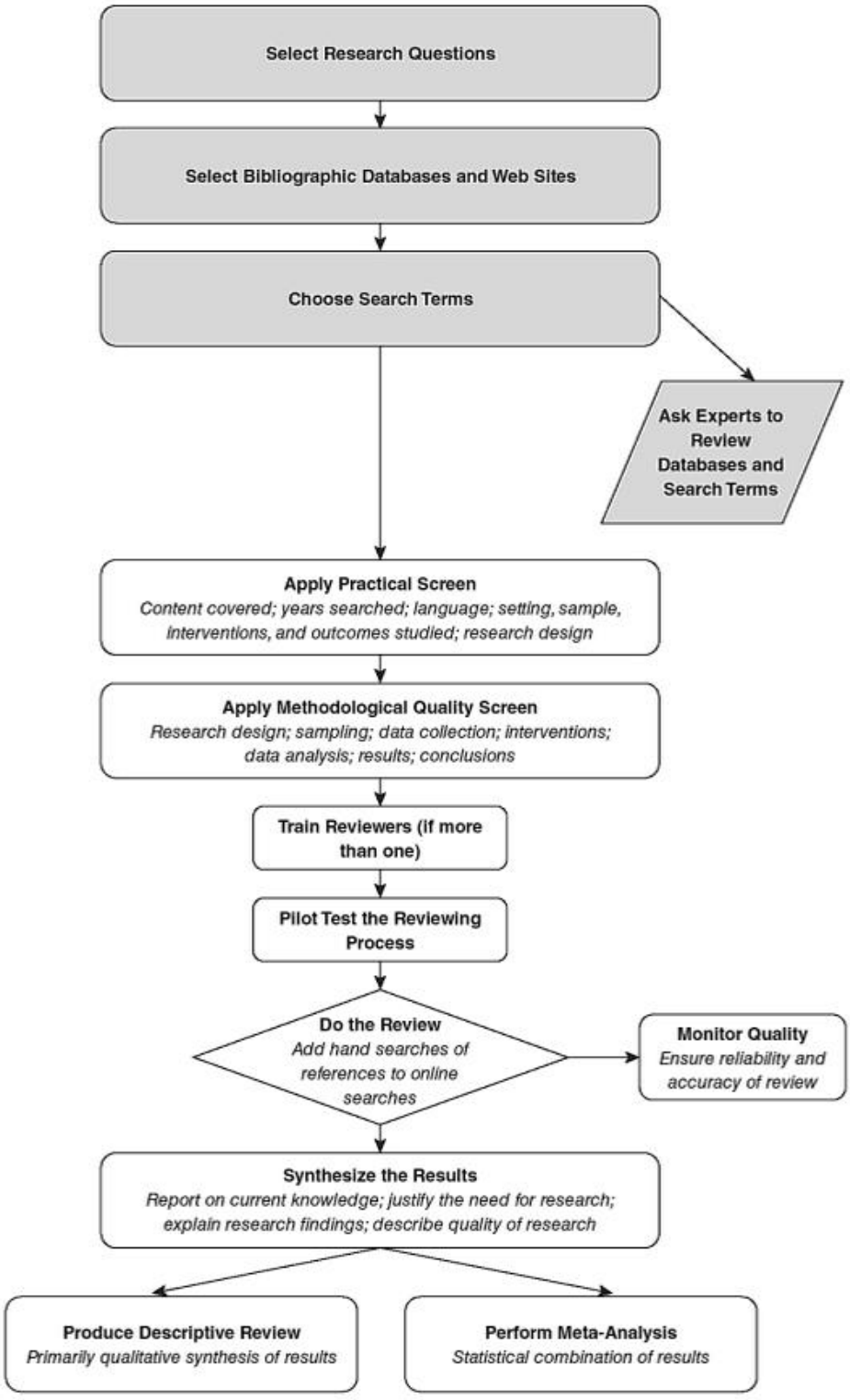
Selkeimmin arkkitehtuurin ja arkkitehtuurimallin eron voi ymmärtää ajattelemalla, että arkkitehtuuri on yksittäinen instanssi arkkitehtuurimallista. Toisin sanoen yhdestä arkkitehtuurimallista voi olla useita erilaisia instansseja arkkitehtuureita.

2.3. Systemaattinen kirjallisuustutkimus

Systemaattinen kirjallisuustutkimus on yksi monista yleisesti hyväksi nähdyistä tavoista toteuttaa tutkimusta. Sen ajatuksena on pyrkiä vastaamaan tutkimuskysymykseen käyttämällä hyväksi olemassaolevaa aineistoa ja sitä kautta pyrkiä muodostamaan uutta tietoa ja uutta tutkimusaineistoa. Systemaattisen kirjallisuustutkimuksen ajatuksena ei ole kuitenkaan hyödyntää kaikkea olemassaolevaa aineistoa, vaan suodattaa siitä tutkimusta varten ainoastaan relevanteimmat aineistot, joiden avulla tutkimuskysymykseen voitaisiin vastata.

Selkeimpänä syynä systemaattisen kirjallisuusselvityksen toteuttamiseen on tiedon pirstaloituminen useiden tutkimusten tuloksena. Kallion [2006] mukaan tiedon ja näkökulmien keskittyminen yksittäisiin julkaisuihin luo ongelman kokonaisuuden näkemisestä; useiden eri tutkimusten keskinäisiä yhteneväisyyksiä on vaikea havaita, ellei aineistoa lähestytä systemaattisesti. Kirjallisuustutkimusten avulla voidaan siis saada kattavampi ja systemaattisempi näkemys aiheesta kuin yksittäisiä julkaisuja tutkimalla, ja aineiston yhtäläisyyksien kautta voidaan päästä kiinni aiheessa selkeimmin pinnalla oleviin asiohin.

Systemaattisen kirjallisuustutkimuksen tulisi aina seurata jotain olemassaolevaa mallia. Tässä työssä pohjana käytettiin Finkin [2005] mallia ja sen tukena Okolinin ja Schabramin [2010] mallia, joka nojaa Finkin malliin. Fink [2005] kuvaa systemaattista kirjallisuuskatsausta metodina, jolla voidaan systemaattisesti, eksplisiittisesti ja toistettavasti identifioida, arvioida ja syntetisoida olemassa olevaa, tutkijoiden ja ammattilaisten tuottamaa tietoa. Finkin [2005] malli on esitetty visuaalisesti kuvassa 1.



Kuva 1. Systemaattinen kirjallisuustutkimus. [Fink, 2005.]

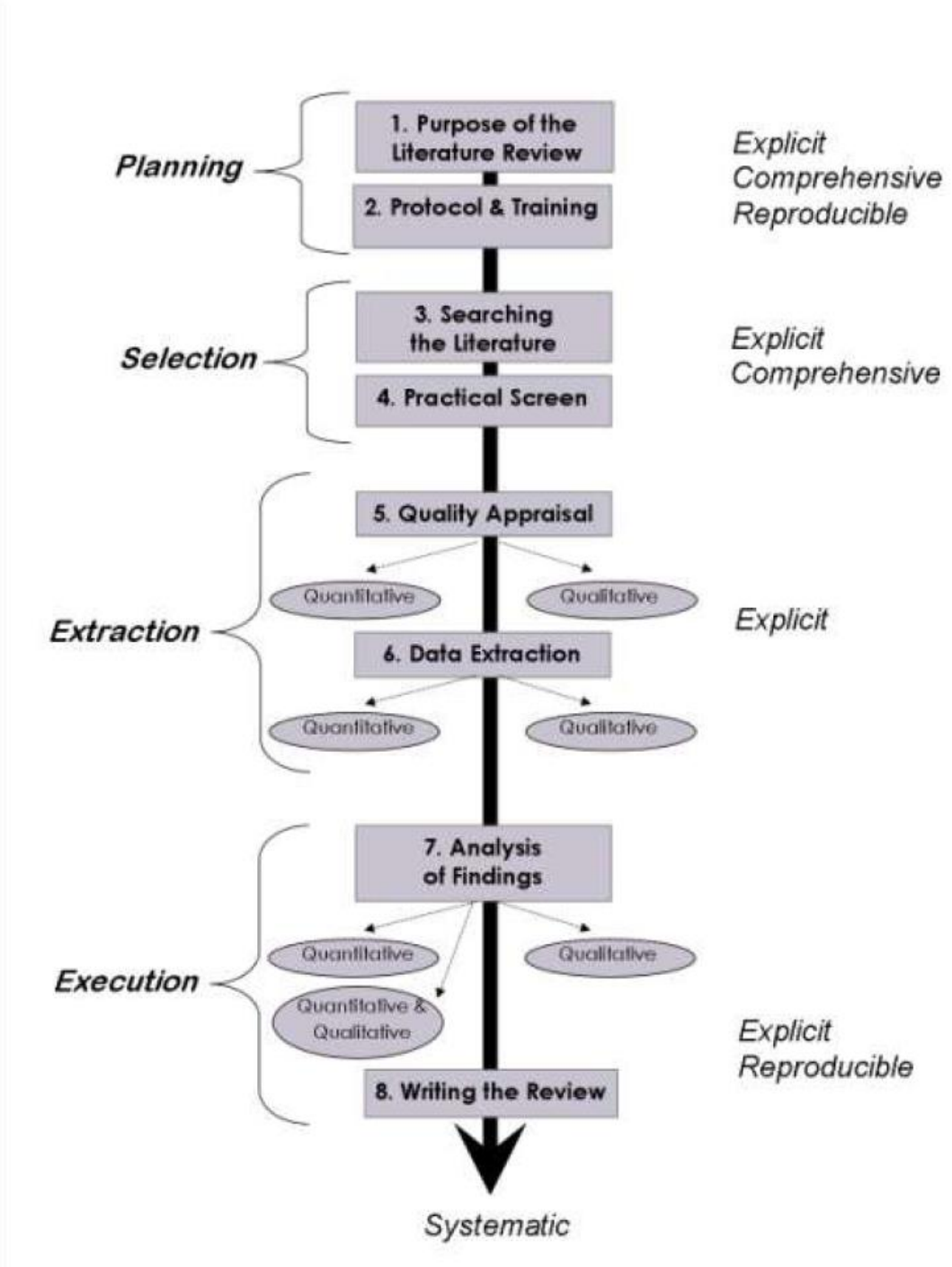
Finkin [2005] mallin ajatuksena ja etuna on sen yksinkertaisuus ja selkeys. Se määrittelee selkeät tehtävät, joiden kautta tutkimusta voidaan lähestyä, ja jakaa ne selkeisiin osakokonaisuuksiin. Näitä osakokonaisuuksia seuraamalla lähes mikä tahansa Finkin [2005] mallia seuraava systemaattinen

kirjallisuusselvitys pitäisi voida toistaa kyseisen työn pohjalta. Finkin [2005] mallin pohjalta voidaan saada aikaan kvalitatiivinen synteesi tuloksista ja statistinen yhdistelmä löydöksistä. Näin ollen se tutkimuksena summaa yhteen löydöksiä kirjallisuudesta ja tuottaa näiden pohjalta uutta tietoa. Tässä työssä Finkin [2005] mallista ohitettiin ainoastaan asiantuntijalausunnot sisällöstä sekä lukijoiden koulutukset ja pilotoinnit. Muulta osin Finkin [2005] menetelmä on kuvattuna ja käytettynä kappaleissa 4 ja 5.

Okolinin ja Schabramin [2010] määritelmän mukaan minkä tahansa kirjallisuusselvityksen lähestymistavan tulee olla yksityiskohtainen ja metodologinen. He kuvaavat kolme eritasoista kirjallisuustutkimusta:

- Teoreettinen taustaselvitys: Tämä osa määrittelee tutkimuksen teoreettisen pohjan ja tuo tutkimuskysymykselle kontekstin sekä fokuksen.
- Kirjallisuusselvitys osana opinnäytetyötä: Ei kokonaisena tutkimuksena, vaan osana esimerkiksi lopputyötä.
- Itsenäinen kirjallisuustutkimus: Artikkelipituinen tutkielma, jonka ainoa tarkoitus on arvioida olemassa olevaa kirjallisuutta analysoimatta mitään muuta aiheeseen liittyvää. Ainoastaan tuloksia julkaisuista tutkimuksista voidaan pitää datana tallaiselle tutkimukselle.

Kun itsenäinen kirjallisuustutkimus toteutetaan käyttäen systemaattista ja standardisoitua menetelmää, voidaan sitä Okolinin ja Schabramin [2010] mukaan kutsua systemaattiseksi kirjallisuustutkimukseksi. Tämän työn osalta on kyse tästä määritelmästä. Kuvasta 2 nähdään, että Okolinin ja Schabramin [2010] malli on hyvin samanlainen kuin Finkin [2005] malli.



Kuva 2. Systemaattinen kirjallisuustutkimus. [Okolin ja Schabram, 2010.]

Mallien lähtökohdat ovat hyvin samankaltaiset, mutta Okolin ja Schabram [2010] tuovat harjoittelun ja protokollien määrittelyt mukaan huomattavan aikaisessa vaiheessa. Tämä voi tehdä tutkimuksen aloittamisesta työläämpää, mutta toisaalta se myös mahdollistaa laadukkaammat tulokset. Okolinin ja Schabramin [2010] malli jakaa sisällön läpikäynnin kahteen osaan, kvalitatiiviseen ja kvantitatiiviseen, niin arvioiden kuin valintojen suhteen. He määrittelevät myös tuotosten

analysoinnin ja tutkimustulosten kirjaamisen huomattavasti Finkiä [2005] tarkemmin ja eksplisiittisemmin. Hart [2005] tarkentaa lisäksi systemaattista kirjallisuustutkimusta seuraavasti:

> *”Kattava kirjallisuustutkimus on tehtävä avoimin mielin ja läpinäkyvästi sen suhteen miten ja miksi aihe valittiin. Lisäksi on oltava avoimia sen suhteen kuinka työn fokus on muuttunut työn aikana tai suhteessa kirjoittajan muuhun työhön tai niiden tuloksiin. Ei myöskään riitä että tutkimus on kokoelma tiivistelmiä muista artikkeleista; sen on sisällettävä myös analyyttistä kritiikkiä.”*

Hartin [2005] ajatus laajentaa vielä aiemmin esiteltyä määritelmää, eli kirjallisuustutkimuksen on kyettävä myös analysoimaan aineistoa ja saavuttamaan päätelmiä, jotka vastaavat tutkimuskyksymykseen. Lisäksi Hart [2005] painottaa tutkijan vastuuta siitä, että hänen tulisi tuoda ilmi seikkoja, jotka ovat vaikuttaneet tutkimuksen suuntaan, jotta kyseiset seikat voidaan huomioida mahdollisissa jatkotutkimuksissa.

3. Tutkimusmenetelmä

Yleensä tutkimuksen menetelmä valitaan sen pohjalta, millaista materiaalia tutkitaan ja millaisia tuloksia halutaan. Tässä tutkimuksessa päädyttiin käyttämään systemaattista kirjallisuustutkimusta, koska kyseisestä aiheesta oli vaihtelevasti aineistoa ja eri aineistot tuntuivat lähestyvän asiaa eri näkökulmista. Systemaattinen kirjallisuusselvitys on looginen väline lähestyä aineistoa ja etsiä sieltä yhteneväisyyksiä ja mahdollista konsensusta, ja sitä kautta lähteä ratkaisemaan tutkimusongelmaa.

Kirjallisuustutkimus on laadullista tutkimusta ja sitä voidaan tehdä monella eri tavalla monen eri menetelmän pohjalta. Tutkimuksen pohjana toimii aiemmin mainittu Finkin [2005] malli, joka koostuu kahdeksasta tehtävästä. Tämä malli valittiin, koska sen tueksi löytyi kirjallisuutta ja sitä oli käytetty muutamassa vastaavassa tutkimuksessa. Finkin mallin tukena käytetään Okolinin ja Schabramin [2010] mallia, joka ottaa kantaa muutamaan Finkin mallin puutteeseen ja tarjoaa joitakin lisäyksiä.

Selkeimpänä syynä laajentaa tutkimusta Okolinin ja Schabramin [2010] mallilla on heidän esittämänsä kritiikki Finkin [2005] mallia kohtaan:

> *”Vaikka Finkin kirja on itsessään ohje jolla voidaan tuottaa kirjallisuustutkimuksia, sen selkeä fokusoituminen terveystieteisiin, ja lähes pinnallinen kvalitatiivisten tutkimusten käsittely, vähentävät sen käytettävyyttä kun tehdään kirjallisuusselvityksiä talouden tai sosiaalitieteiden parissa.”*

Koska Finkin [2005] mallia on kuitenkin käytetty ahkerasti mm. informaatiotutkimuksen puolella, katsottiin, että se soveltuu tämän tasoiseen tutkimukseen paremmin kuin pelkästään Okolinin ja Schabramin [2010] mallin käyttäminen.

Suurimpana heikkoutena Finkin [2010] menetelmän käytössä voidaan nähdä sen soveltajan subjektiivisten valintojen vaikutus tutkimukseen. Sekä aineiston että hakutermien suhteen oli mahdollista tehdä puhtaasti subjektiivisia valintoja, jotka ohjasivat tulosta mahdollisesti liikaakin haluttuun suuntaan. Tästä syystä Hart [2005] korostaakin tutkijan subjektiivisten valintojen vaikutuksia ja painottaa tutkijan vastuuta kirjata työhön tekijät, jotka vaikuttivat valintoihin ja suhtautumiseen työhön.

3.1. Tutkimuskysymykset

Tutkielman alkuperäinen tarkoitus oli luoda kevyt kirjallisuuskatsaus e-oppimisjärjestelmien arkkitehtuurimalleihin. Tätä katsausta oli tarkoitus käyttää osana laajempaa lopputyötä, mutta siitä muodostui lopulta niin laaja ja kattava, että peilaten Okolinin ja Schabramin [2010] määritykseen itsenäisestä kirjallisuusselvityksestä nähtiin, että katsaus muodostui jo itsessään tutkielmaksi.

E-oppimisjärjestelmät ovat nykypäivänä erittäin käytettyjä, ja valmiita ratkaisuja on tarjolla useita. Vaikka järjestelmät on usein mahdollista hankkia valmiina paketteina, niiden käyttöönotto ei silti ole täysin yksiselitteistä, ja tähän liittyykin tutkielman kysymyksenasettelu.

Arkkitehtuurin tarkoitus on kuvata, miten ja millä säännöillä järjestelmä tulisi rakentaa tai ottaa käyttöön. Nämä säännöt voivat olla mukana jo kun ratkaisua ollaan määrittämässä, ja/tai rakentamassa. Tämän vuoksi onkin tärkeätä ymmärtää, millaista tutkimusta aiheesta on tehty ja millaisiin tuloksiin niissä on päädytty. Tätä tietoa voidaan hyödyntää suoraan uusia järjestelmiä hankittaessa tai jatkotutkimusaiheita mietittäessä.

Tutkimuskysymykseksi muodostuikin siis:

”Millaisia malleja e-oppimisarkkitehtuuriin löytyy kirjallisuuden perusteella?”

Koska suomenkielistä materiaalia on varsin rajoitetusti, tutkimuskysymys käännettiin myös englanniksi, jotta voitiin käyttää sitä osana tutkimuksen toteutusta hakemalla materiaalia laajemmalta. Vapaasti käännettynä se oli:

”What kind of architectural models are there for e-learning systems?”

Näiden kysymysten pohjalta lähdettiin toteuttamaan varsinaista tutkimusta.

4. Tutkimuksen toteutus

Tutkimuksen toteutuksessa lähdettiin liikkeelle Finkin [2005] mallin kahdeksan kohdan avulla:

1. Valitaan tutkimuskysymykset
2. Valitaan tiedonlähteet

3. Valitaan hakutermit
4. Asetetaan ensimmäinen (käytännön) seula ja seulotaan
5. Asetetaan toinen (metodologinen) seula ja seulotaan
6. Suoritetaan katsaus
7. Syntetisoidaan tulokset
8. Arvion / päätelmien kirjoittaminen

Okolin ja Schabram [2010] esittivät myös oman kahdeksan kohdan mallin:

1. Kirjallisuustutkimuksen tarkoituksen määrittely
2. Osallistujien kouluttaminen ja käytäntöjen määrittely
3. Kirjallisuuden hakeminen
4. Käytännön seula
5. Laatuseula
6. Oleellisen tiedon noutaminen aineistosta
7. Aineiston analysointi
8. Arvion / päätelmien kirjoittaminen

Okolinin ja Schabramin [2010] malli lähtee liikkeelle työn tarkoituksen kautta aivan kuten Finkinkin [2010] malli, mutta ottaa heti alussa kantaa tekijöiden koulutukseen ja käytäntöjen määrittelyyn (tosin tämä on mukana myös Finkillä, mutta vasta myöhemmässä vaiheessa). Okolin ja Schabram [2010] eivät myöskään määrittele aivan yhtä tarkkoja menetelmiä kirjallisuuden valintaan Finkiin verrattuna [2005]. Muuten seulojen käytännöt eroavat lähinnä nimellisesti; vaikutukset aineistoon ovat käytännössä samat. Okolin ja Schabram [2010] erittelevät vielä loppuosaan erikseen tiedon poiminnan aineistosta, minkä Fink [2005] lukee sisälle katsauksen toteuttamiseen. Molemmat mallit päättyvät arvion / päätelmien kirjoittamiseen.

Ensimmäinen kohta eli tutkimuskysymyksen määrittäminen ja tutkimuksen perustelu on kuvattu aikaisemmissa kappaleissa, joten käytännön toteutuksen kannalta tiedonlähteiden valinta oli seuraava toiminnallinen kohta.

4.1. Tiedonlähteet

Finkin [2005] mukaan tiedonlähteiden tarkoitus on auttaa löytämään tutkimusaineistoa, joka vastaaa tutkimuskysymykseen. Mallissa keskitytään pääosin tietokannoista tehtyihin tiedonhakuihin, joten tiedonlähteinä pyrittiin käyttämään pääasiallisesti niitä.

Okolin ja Schabram [2010] määrittelevät, että aineiston valinta sekä hakujen toteutus on määriteltävä eksplisiittisesti ja pystyttävä perustelemaan. Käyttäen apuna Tampereen yliopiston kirjaston henkilökuntaa (kuten Okolin ja Schabram [2010] suosittelevat) päädyin käyttämään lähteenä tietokantojen valinnalle Tampereen Yliopiston kirjaston Tiedonlähteet-sivua [Tampereen Yliopisto, 2015]. Lopulliset tietokantavavalinnat pohjasivat subjektiivisiin kokemuksiin tietokantojen sisällöstä ja kattavuudesta eri tieteenalojen osalta sekä niiden sisältämän tiedon määrästä. Vaikuttavana tekijänä oli myös pääsy tietokantoihin yliopiston ulkopuolelta.

Tietokannoista valittiin tutkimukseen seuraavia:

Kotimaisista tietokannoista:

1. Tamcat
2. Arto
3. Melinda

Kansainvälisistä tietokannoista:

1. Science Direct
2. EBSCOhost (Academic Search Premier)
3. SpringerLink
4. LISA

4.2. Hakutermien valinta

Finkin [2005] mukaan hakutermien ja -lausekkeiden tulisi kuvastaa kysymyksiä niin, että niiden avulla saataisiin mahdollisimman relevantteja tuloksia. Tällöin yhdenkin harkitun ja eksaktin tutkimuskysymyksen pohjalta voidaan muodostaa riittävästi hakutermejä. Myös tarkka rajaus on oleellista, koska liian sallivan rajauksen myötä hakutulosten määrä kasvaa helposti yli järkevästi käsiteltävän rajan.

Hakutermeissä on siis pyritty käyttämään omaa ymmärrystä relevanteista hakutermeistä ja rajattu termejä jo hieman sen perusteella, millaisilla hakutermeillä oman käsityksen mukaan voisi päätyä

laadukkaisiin tuloksiin. Kuitenkin on pyritty pitämään mielessä, että hakutermien tulisi hakea tarpeeksi kattavasti erilaista materiaalia lähteistä sekä suomeksi että englanniksi.

Suomenkieliset hakutermit perusmuodossa:

- e-oppiminen [1]
- verkko-opetus [1]
- e-opetus [1]
- arkkitehtuuri [2]
- arkkitehtuurimalli [2]

Englanninkieliset hakutermit perusmuodossa:

- e-learning[1]
- elearning [1]
- architecture [2]
- architecture model [2]

Hakutermejä yhdistettiin Boolen operaattoreilla AND ja OR. Yhdistäminen tapahtui käyttämällä ykköstermejä ja kakkostermejä ([1],[2]) esim. ”e-oppiminen AND arkkitehtuuri” ja ”elearning AND architecture model”. Yhdistelemällä termejä käyttäen sekä ykköstermejä että kakkostermejä yhdessä varmistettiin, että haut koskettivat sekä arkkitehtuuria että e-oppimista, eikä esimerkiksi vain arkkitehtuuria ja arkkitehtuurimallia.

4.3. Ensimmäinen seulontavaihe

Pelkkiä hakutermejä käyttäen saataisiin valtava määrä materiaalia, jota ei olisi mahdollista käsitellä järkevässä ajassa. Tämän vuoksi Fink [2005] suosittelee kahta seulontavaihetta, ensimmäistä käytännön seulaa, ja toista metodologista seulaa. Okolin ja Schabram [2010] kuvaavat näitä seuloja nimillä ”Käytännön seula” ja ”Laatuseula”, mutta tarkoituksiltaan molempien mallien seulat ovat hyvin samanlaiset.

Ensimmäisen seulontavaihteen kriteerit voivat liittyä esimerkiksi kieleen, ajankohtaan, julkaisuun tai tutkimusasetelmaan. Varsinaisten kriteerien valintaan ei ole yhtä oikeaa tapaa, vaan se riippuu tutkimuksen tarpeista ja resursseista. Ensimmäiseen seulontavaiheeseen sisältyy myös hieman riskiä, sillä hakusanat ja kriteerit valitaan subjektiivisesti. Liian tiukkaan valitut kriteerit vaikuttaisivat löydetyn materiaalin määrään ja laatuun, kun taas liian sallivat kriteerit tuottaisivat liian suuren määrän aineistoa, jota ei olisi mahdollista käsitellä. Tämä riski kuitenkin tiedostettiin ja pyrittiin

toimimaan sen puitteissa niin, että kriteerit ja hakusanat olisivat riittävän oikeellisia tutkimuskysymystä silmällä pitäen.

Ensimmäisen seulontavaiheen kriteereiksi valittiin seuraavat tekijät:

- *Julkaisun tulee käsitellä e-learningia nimenomaan arkkitehtuurinäkökulmasta, tai julkaisussa tulee selkeästi tulla esille arkkitehtuurinäkökulma*. Tämä tulee esille jo hakutermin valinnassa, ja tutkimuskysymyksen asettelussa.
- *Aineisto on joko suomen tai englannin kielellä.* Kielivalinta on laajennettu koskemaan myös englannin kieltä, koska pelkästään suomenkielinen materiaali ei ollut tarpeeksi kattavaa.
- *Aineisto on julkaistu 2004–2014.* Jotta materiaali olisi relevanttia ja ajantasaista, päädyin rajaamaan aineiston viimeiselle vuosikymmenelle.
- *Aineiston tulee olla laadukas.* Laadukkaalla tarkoitetaan, että julkaisu on vähintään pro gradu -tasoinen ja mielellään vertaisarvioitu.

Näiden kriteerien pohjalta ensimmäisen seulontavaiheen haku suoritettiin 8.11.2014, ja niiden pohjalta löytyi aineistoa seuraavasti:

Kotimaiset tietokannat (käytetty ainoastaan suomenkielisiä hakutermejä):

- Tamcat: 0
- Arto: 0
- Melinda: 1

Yhteensä: 1 kpl.

Kansainväliset tietokannat (käytetty ainoastaan englannikielisiä hakutermejä):

- ScienceDirect: 118685
- EBSCOhost (Academic Search Premier): 25
- Springerlink: 12261
- LISA: 77

Hakutulosten osalta ei ole eroteltu julkaisuja, jotka olivat useassa eri hakutuloksessa, koska ristiinvertailua oli materiaalin laajuuden pohjalta mahdotonta tehdä. ScienceDirectin ja Springerlinkin osalta käytiin läpi relevanssin mukaan järjesteltynä 100 ensimmäistä tulosta. Suuremman määrän käsittely olisi ollut mahdotonta järkevässä ajassa. Yhteensä läpikäytäviä

aineistoja ensimmäisessä vaiheessa oli siis 302 kappaletta. Nämä aineistot käytiin läpi otsikkotasolla ja valittiin relevanteimmilta vaikuttavat julkaisut (15 kpl) läpikäytäväksi tiivistelmätasolla.

Otsikkotasolla käsiteltyjä aineistoja ei erikseen listattu lähteisiin noudattaen Finkin [2005] ohjeistusta siitä, että otsikkotason käsittelyssä aineisto tulisi käydä läpi nopeasti, koska määrä voi olla hyvin suuri. Tältä osin tutkimuksen toistaminen voi olla ongelmallista, koska tietokantahakujen tulokset elävät ajan kanssa ja mahdolliset uudet haut samoilla hakutermeillä voivat antaa eri ajankohtina eri tuloksia. Tämän vuoksi onkin tärkeää ymmärtää otsikkotason käsittelyn metodologia ja kriteerien asettamisen merkitys. Kun nämä on huomioita, lopputuloksen ei pitäisi erota oleellisesti eri tutkimusajankohtien välillä.

Otsikkotason läpikäynti ei välttämättä antanut oikeellisinta kuvaa julkaisusta, mutta tarkempi tarkastelu olisi ollut ajankäytöllisesti mahdotonta. Tiivistelmätason läpikäynti oli myös hieman ongelmallista, koska aineiston laajuudesta ja erilaisuudesta johtuen kaikissa aineistoissa ei ollut yhtäläistä tiivistelmää tai abstraktia. Osa aineistosta jouduttiin karsimaan puutteellisen tiivistelmätiedon pohjalta. Yhden löydetyn aineiston osalta koko tekstin lataaminen ei onnistunut, joten se piti hylätä.

Tiivistelmätason pohjalta valittiin toiseen seulontavaiheeseen kuusi artikkelia.

4.4. Toinen seulontavaihe

Toisen seulontavaiheen kriteerit perustuvat tutkimusten sisältöön, tekotapoihin, tuloksiin ja johtopäätöksiin. Toisen seulontavaiheen tarkoituksena on tiputtaa aineistosta pois ne julkaisut, jotka eivät vastaa tutkimuskysymykseen tai eivät ole muuten relevantteja tutkimukselle. Artikkelissa voidaan esimerkiksi käsitellä tutkimuksen kannalta kiinnostavia asioita, mutta mikäli käsitellyt asiat eivät selkeästi vastaa tutkimuskysymykseen, kyseinen artikkeli tulisi jättää pois loppumateriaalista. Kaikki artikkelit, joissa viitattiin valmiisiin toteutuksiin, karsittiin tämän työn osalta tässä vaiheessa pois, koska haluttiin keskittyä arkkitehtuurivalintoihin, ei arkkitehtuurin pohjalta tehtyihin valintoihin.

Finkin [2005] mukaan toisen seulontavaiheen jälkeen tulisi jäljellä olla enää niitä tutkimuksia, joita halutaan lopulliseen kirjallisuuskatsaukseen, eli aineistoa, jota analysoidaan tarkemmin ja jonka pohjalta tehdään päätelmät. Toisen seulontavaiheen kriteereillä aineistosta tippui pois yksi julkaisu, ja loput jäljelle jääneet viisi lähdettä luettiin kokonaisuudessaan.

4.5. Katsauksen suorittaminen

Katsaus perustuu jälleen Finkin [2005] malliin. Finkin mukaan katsauksen tulisi olla systemaattinen ja siinä tulisi analysoida aineistoa sillä tasolla, että se tuottaisi korkealaatuista tutkimusmateriaalia. Sen avulla tulisi löytää aineistosta selkeitä johtopäätöksiä, joita voidaan käyttää tutkimuskysymyksiin

vastaamiseen. Katsausvaiheessa arvioijalla on seulontavaiheiden jäljiltä käytössään enää ne aineistot, joiden pohjalta päätelmät tullaan kirjoittamaan. Näiden aineistojen sisältä tulisi systemaattisesti poimia informaatiota, jonka pohjalta syntetisointi voitaisiin toteuttaa [Okolin ja Schabram, 2010]. Katsauksen suorittamisen tarkoituksena on vastata niihin kysymyksiin, joita tutkimuksen pohjaksi on valittu, käyttäen hyväksi tarkoin valikoitunutta ja laadukasta aineistoa.

Tutkimustulokset käsitellään erikseen luvussa 5.

4.6. Tutkimustulosten syntetisointi

Kun kaikki aineisto on läpikäyty, suodatettu, valittu ja arvioitu, on vuorossa tutkimustulosten syntetisointi. Syntetisoinnin tarkoituksena on yhdistää aineisto yhdeksi kattavaksi katsaukseksi, johon tiivistyy usean eri artikkelin sisältö. Tässä kohtaa katselmoijan olisi Okolinin ja Schabramin [2010] mukaan tarkoitus koota, organisoida ja vertailla aineistoa ja keskustella siitä.

Tutkimustulosten syntetisointi käsitellään luvussa 6.

5. Tutkimustulokset

Finkin [2005] mukaan kirjallisuustutkimuksen tulisi käsittää aiheen nykytila, uuden tutkimuksen tarve, löydökset ja aineiston taso. Tämä kappale käsittää nykytilan, löydökset ja aineiston tason. Uuden tutkimuksen tarve läpikäydään kappaleessa 6.4.

Havaintojen perusteella tutkimustulokset jaoteltiin viiteen eri kategoriaan:

- Aineistoon liittyviin
- Standardeihin liittyviin
- Modulaarisuuteen liittyviin
- Muihin malleihin liittyviin
- Teknologiavalintoihin

Nämä tasot käydään läpi omina kokonaisuuksinaan, joiden tarkoitus on avata kategoriaan liittyviä näkökulmia ja löydöksiä. Jaottelun taustalla on Kallion [2006] näkemys tiedon pirstaloitumisesta useisiin eri tutkimuksiin. Jaottelemalla löydökset selkeisiin kategorioihin saadaan nämä pirstaloituneet tiedot takaisin kokonaisuuksiin ja sitä kautta luodaan parempi ymmärrys kyseisistä aiheista.

5.1. Aineisto

Kirjallisuustutkimuksen tuloksissa on selkeintä lähteä liikkeelle aineistosta löytyineistä yhteneväisyyksistä. Aineiston ristiinanalysointi ja vertailu antaa hyvän kuvan aihealueen kypsyydestä tai sen puutteesta. Mikäli yhteneväisiä artikkeleja löytyy paljon, se kertoo aiheen kypsyydestä ja konsensuksesta kirjoittajien välillä, kun taas vastaavasti yhteneväisyyksien vähyys kertoo kypsyyden puutteesta tai ristiriidoista kirjoittajien välillä.

Kaikki artikkelit käytiin läpi kokonaisuudessaan ja niiden avulla pyrittiin löytämään vastausta asetettuun tutkimuskysymykseen. Yllättävänä asiana materiaaleista paljastui, että kukaan ei ollut suoraan verrannut erilaisia arkkitehtuurimalleja keskenään tai muutenkaan tehnyt kattavaa katsausta erilaisiin käytettyihin arkkitehtuurimalleihin tai arkkitehtuureihin. Käytännössä aineistossa oli esitelty erilaisia arkkitehtuurimalleja e-oppimisjärjestelmien toteuttamiseksi ja perusteltu, miksi juuri kyseinen arkkitehtuurimalli olisi paras valinta toteutukselle.

Useissa artikkeleissa ei juurikaan tehty vertailuja oman mallin ja kilpailevien mallien välillä, vaan lähinnä perusteltiin oman mallin hyviä puolia ja sen soveltuvuutta ko. toteutukseen. Osa artikkeleista meni myös toteutuksen kannalta hyvin teknisiin yksityiskohtiin, jotka eivät tämän tutkimuksen kannalta olleet oleellisia, minkä vuoksi sopivia artikkeleita tutkimuskysymyksen kannalta oli neljä.

Osa materiaalista ei varsinaisesti ollut tutkimusta, vaan kuvasi enemmänkin kirjoittajien omia mielipiteitä, eikä selkeää vertailevaa tutkimusta ollut suoritettu. Koska materiaalia oli kuitenkin niukasti, hyväksyttiin myös ammattilaisten omat ajatukset osaksi tutkimusaineistoa. Aineistoon otettiin varsinaisten seulontojen ulkopuolelta mukaan vielä yksi lähde liittyen ”IEEE Learning Technology Standards Committee”:n julkaisuihin, sillä siihen viitattiin useassa artikkelissa ja se koettiin relevantiksi tutkimuskysymyksen kannalta.

Yleisesti ottaen artikkeleissa oli hyödynnetty sekä ADL:n, IEEE:n että IMS:n standardeja, ja näiden päälle oli pyritty löytämään parannuksia. Ne toimivat lähtökohtina monissa eri tutkimuksissa ja toteutuksissa, ja siten kuvaavat sitä pohjaa, jota e-oppimisjärjestelmille nykypäivänä on olemassa. Selkeitä yhteisiä viittauksia löytyi hyvin vähän eikä selkeitä pohjateoksia havaittu aineistossa, vaan jokainen kirjoittaja viittasi hyvin valikoidusti lähinnä omia näkemyksiään tukeviin aineistoihin. Toisaalta tämän työn puitteissa ei voitu tehdä kattavaa ristiinvertailua koko aineiston lähteistä, mikä olisi voinut paljastaa enemmän yhteisiä lähteitä.

5.2. Standardit ja mallit

Standardit ja mallit valittiin yhdeksi kokonaisuudeksi, sillä lähes jokainen artikkeli viittasi johonkin olemassaolevaan standardiin tai malliin. Voidaankin nähdä, että aihealueella on olemassa jonkinlaista

kypsyyttä ainakin standardien ja mallien osalta, koska yksikään kirjoittajista ei ollut rakentanut malliaan ilman, että olisi pohjannut sitä johonkin aikaisempaan malliin tai standardiin.

Standardeja avataan Duanin *et al*. [2006] artikkelissa "An Architecture for Online Laboratory E-learning System", jossa käsitellään mm. historiaa standardien taustalla. Artikkeli listaa historiaa e-oppimisjärjestelmien taustalla kronologisesti siinä määrin kun kirjoittajat ovat sitä löytäneet. Ensimmäisiä standardeja e-oppimisjärjestelmille on artikkelin mukaan valmisteltu "Aviation Industry CBT Committee":n toimesta jo vuonna 1988. Tällöin on julkaistu useita suosituksia mm. ohjelmiston ja raudan suhteen. Tämän jälkeen perustettu "IMS Global Learning Consortium" on julkaissut mm. ohjeita, kuinka XML-pohjaista tiedonvaihtoa voitaisiin hyödyntää e-oppimisjärjestelmissä.

Vuonna 1996 perustettiin "IEEE Learning Technology Standards Committee" (LTSC) tavoitteenaan kehittää ja julkaista alan standardeja. LTSC:n ehkä tunnetuin tuotos on "Learning Object Metadata" -spesifikaatio, joka määrittelee oppimisresursseja kuvaavat elementit ja elementtiryhmät. Tätä LOM-spesifikaatiota avataan lisää myöhemmin. LTSC on lisäksi määritellyt myös "Learning Technology Standard Architecture (LTSA)" -mallin, jota avataan myöhemmin lisää.

Viimeisimpänä määritelmiä julkaisi vuonna 1997 "Advanced Distribution Learning" -aloite (ADL), joka oli Valkoisen talon teknologiaosaston aikaansaannos. Tämän aloitteen pohjalta määriteltiin mm. "Shareable Courseware Object Reference Model"-malli (SCORM), joka on käytössä nykypäivänäkin ja johon muutamat artikkeleissa mainitut järjestelmät pohjaavat.

Sekä aiemmin mainittu IMS että ADL käyttävät hyväksi IEEE:n LOM:n elementtejä ja rakenteita.

Standardeja ja malleja on hyödynnetty monissa eri artikkeleissa, ja esimerkiksi Chon ja Leen [2004] artikkelissa "A Module-Based Software Framework for E-learning" viitataan IEEE:n "Learning Technology Standard Architecture" -malliin. Kyseinen malli määrittelee viisi eri tasoa e-oppimisjärjestelmälle:

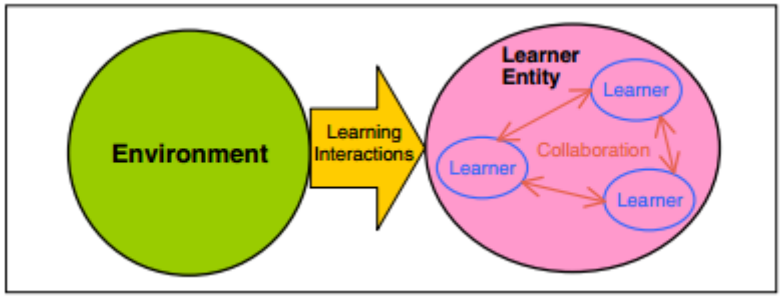
Taso:

1. Learner / Enviromental Interactions
2. Learner-Related Design Features
3. IEEE 1484.1 LTSA System Components
4. Stakeholder Perspectives/Prioritites
5. Codings, APIs, & Protocols

Kyseinen IEEE:n standardi määrittelee ne tasot, joiden pohjalta e-oppimisjärjestelmiä tulisi rakentaa. Jokainen taso sisältää eri määrän asioita, joita tulee ottaa huomioon kyseisen tason suunnittelussa.

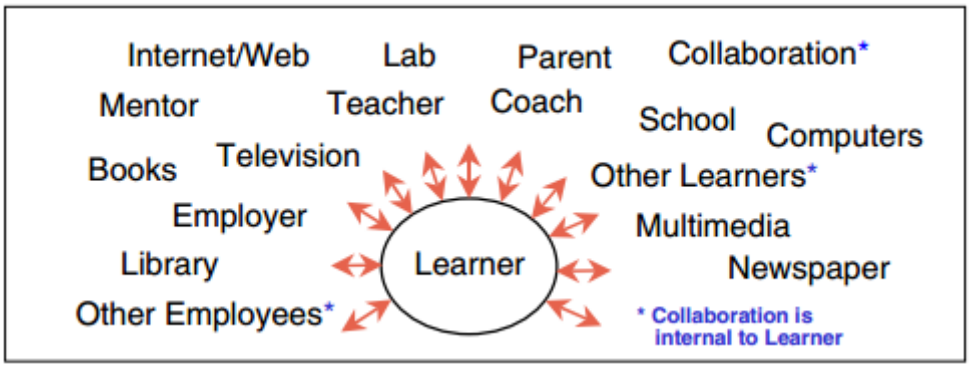
Kyseiset tasot menevät suunnittelun kannalta tärkeysjärjestyksessä eniten tärkeästä vähiten tärkeään. Huomionarvoista on, että varsinainen toteutus on hyvin pienellä tärkeysasteella, ja selkeästi enemmän panostusta tulisi laittaa ylemmän tason suunnittelutehtäviin kuin varsinaiseen tekniseen toteutukseen.

Ensimmäinen taso (kuva 3) määrittelee varsinaisen oppimisen eli sen, kuinka käyttäjä löytää, vaihtaa, formuloi, siirtää, jne. tietoa ja/tai informaatiota suhteessa ympäristöön. Tason tarkoituksena on siis kuvata varsinainen tiedon välitys (eli oppiminen).



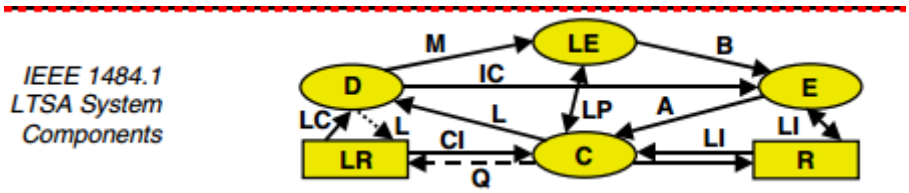
Kuva 3. Oppiminen. [IEEE, 2003.]

Toinen taso (kuva 4) määrittelee suhteet, joita oppijalla on järjestelmän suunnitteluun – käytännössä siis kaikki sidosryhmät ja toimijat, joita järjestelmän suunnittelussa pitää ottaa huomioon.



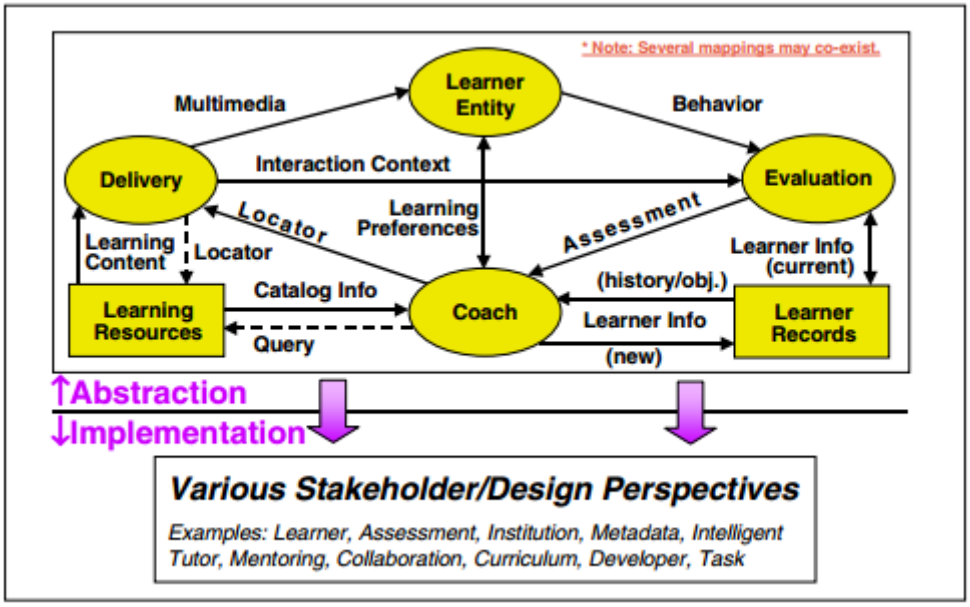
Kuva 4. Suhteet. [IEEE, 2003.]

Kolmas taso (kuva 5) määrittelee teknologiasidokset ja erilaiset rajapinnat, joita tulisi ottaa huomioon. Tässä huomioidaan myös järjestelmän sisäiset komponentit ja niiden väliset tiedonvaihdot.



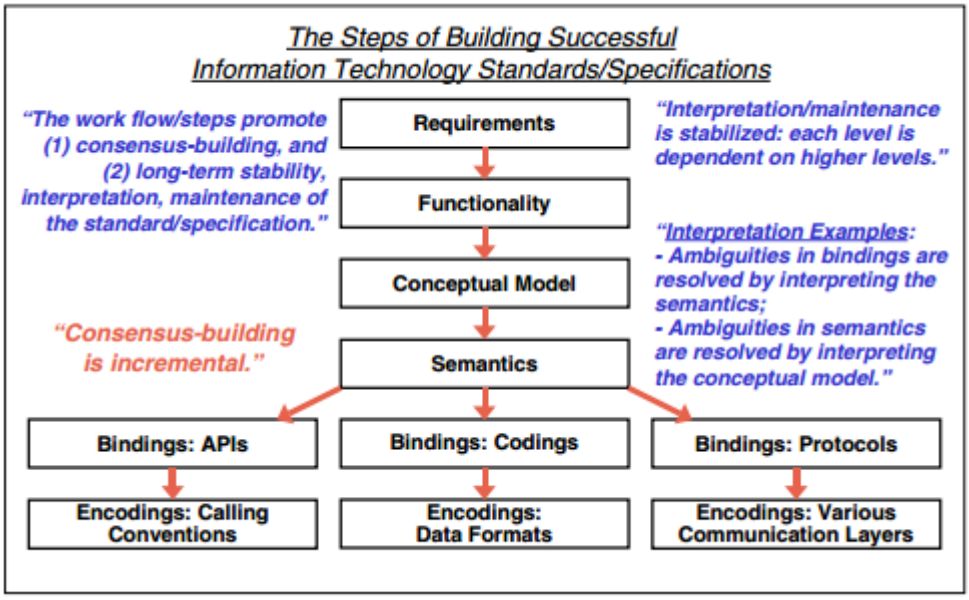
Kuva 5. Teknologiasidokset [IEEE, 2003.]

Neljäs taso (kuva 6) määrittelee jälleen erilaisia sidosryhmiä ja tapoja niiden määrittämiseen. Standardi määrittää yli 20 erilaisia tapausta ja lähtökohtaa, joiden näkökulmasta sidosryhmiä voidaan miettiä. Erilaisia näkökulmia ovat mm. “Learner-centered”, ”Assessment-centered”, ja “Student Administration systems”.



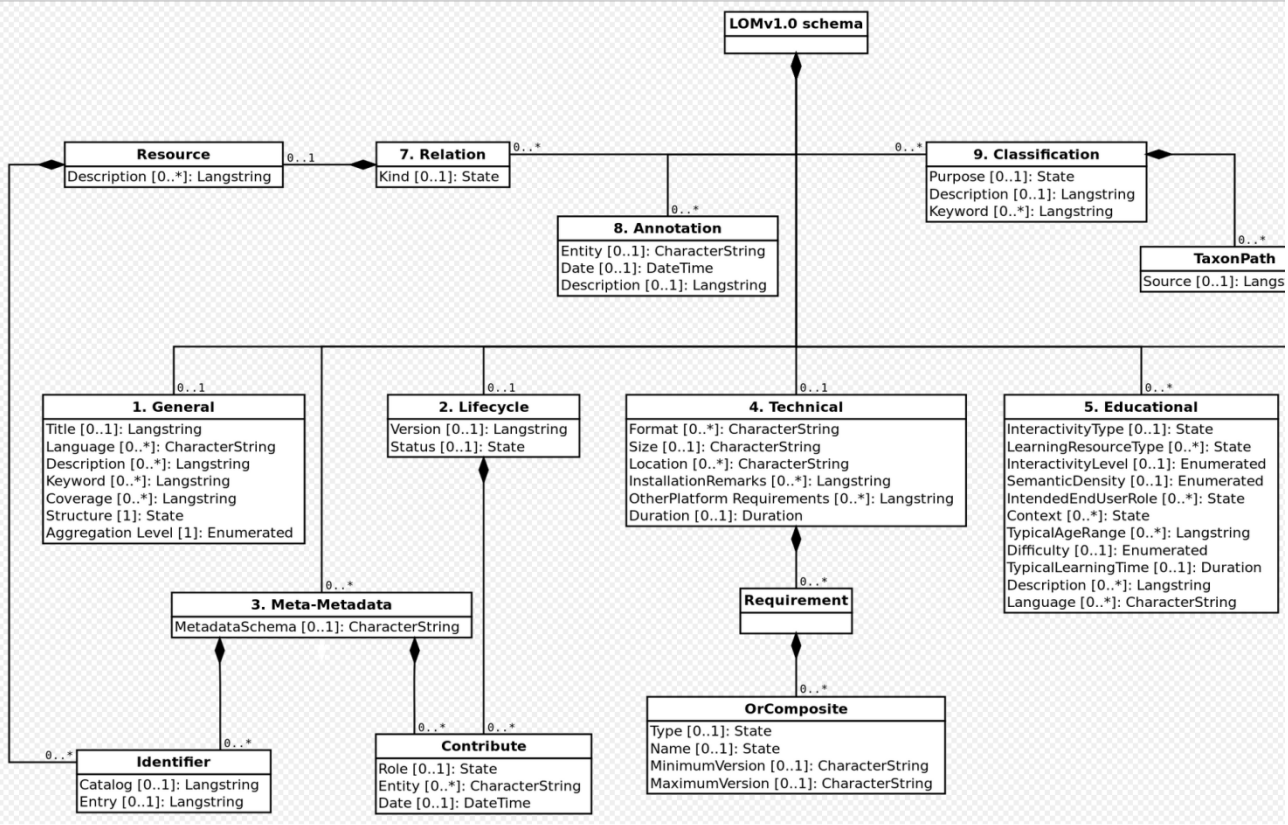
Kuva 6. Sidosryhmät. [IEEE, 2003.]

Viides taso (kuva 7) määrittelee lähtökohtia teknologiatoteutuksille ja rajapinnoille. Tällä tasolla varmistetaan, että järjestelmässä on riittävästi rajapintoja toteutettuina järkevillä teknologioilla ja että sen toteutus modulaarisena sekä laajennettavuus on riittävän hyvin huomioitu.



Kuva 7. Rajapinnat ja koodi. [IEEE, 2003.]

IEEE [2003] määrittelee arkkitehtuurin lisäksi myös ”Learning Object Metadata” –mallin (kuva 8), jonka avulla e-oppimisjärjestelmien rakentajat voivat määritellä tietojen kuvaukset. ”Learning object metadata” on yleensä XML:llä toteutettu datamalli, joka kuvaa oppimisen tukena käytettävän digitaalisen resurssin. Mallin ideana on, että resursseja voitaisiin uudelleenkäyttää ja hyödyntää mahdollisimman laajalti. Yleisimmin tällaisia resursseja käytetään juuri e-oppimisjärjestelmissä.



Kuva 8. Learning object metadata. [Wikipedia, 2014.]

Varsinainen “Learning Object” on kuvaus oppimistapahtumasta, yleensä kuvattuna LOM:n määrittelemällä tavalla XML-muodossa. Ajatuksena on, että ”Learning Object” olisi kuvattu oppimistapahtuma, jolle olisi määritelty kaikki siihen liittyvät ominaisuudet niin, että se olisi uudelleenkäytettävissä mahdollisimman helposti. Esimerkiksi kurssi, jonka sisältö ei vaihdu vuosittain, voisi olla ”Learning Object”.

Esimerkkinä yleisimpiä asioita, joita ”Learning Objectien” kuvauksissa (ja metadatoissa) on:

- Kurssin kuvaus (lyhenne, kieli, aihealue, kuvaus, avainsanat)
- Elinkaari (versio, tila)
- Lisätiedot (weblinkit, kuvat, ääni, video)
- Terminologia
- Käyttöoikeudet
- Suhteet muihin kursseihin

Näitä ominaisuuksia ja datamalleja hyödyntämällä saadaan kuvattua kursseja oppimisjärjestelmiin niin, että ne ovat standardoituja, uudelleenkäytettäviä, ja sisällöllisesti oikeellisia.

5.3. Modulaarisuus

Modulaarisuus oli yksi selkeimpiä yhtäläisyyksiä eri kirjoittajien välillä. Lähestulkoon jokainen kirjoittaja oli jollain tavalla määritellyt arkkitehtuurinsa modulaariseksi. Ainoastaan se, miten modulaarisuus määriteltiin, ja mitä kaikkea arkkitehtuurista koettiin modulaarisena, vaihteli kirjoittajalta toiselle. Esimerkkinä Duanin *et al*. [2006] artikkeli ”An Architecture for Online Laboratory E-learning System” kuvaa yleisen kaavan tyypilliselle e-oppimisjärjestelmälle seuraavasti:

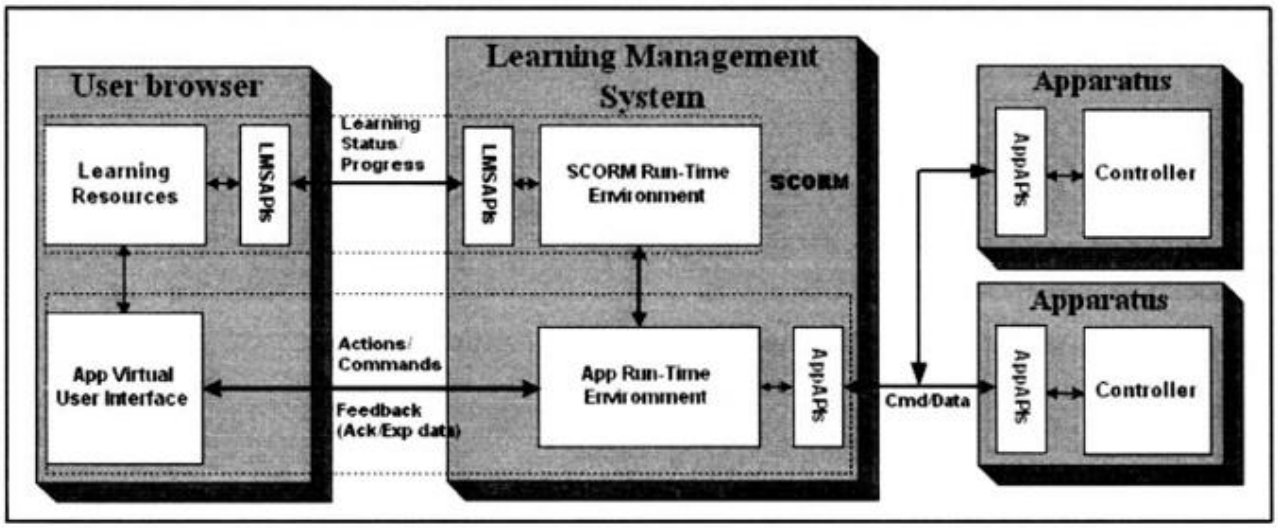
LMCS = LMS + CMS [RLOs]

LMCS eli ”Learning management content system” on monen käyttäjän järjestelmä, jossa käyttäjät voivat järjestää, hallita ja jakaa sisältöä keskitetysti.

LMS on ”Learning Management System”, jonka tarkoituksena on hallita keskitetysti opiskelijoita ja oppimiseen liittyviä asioita.

CMS eli ”Content management system” on järjestelmä, jossa hallitaan sisältöä, kuten kuvia ja tekstejä. Esimerkiksi monien verkkosivustojen taustalla on CMS-järjestelmä, jonka avulla hallitaan sisältöä erillään teknisestä toteutuksesta. CMS pitää tässä kontekstissa usein sisällään RLO:ita, ”Reusable learning objects”, eli uudelleenkäytettäviä objekteja. Nämä IEEE:n LOM-standardiin pohjaavat objektit ovat standardoituja kuvauksia kokonaisuuksista, joita voidaan hyödyntää uudelleen ja uudelleen ja eri konteksteissa.

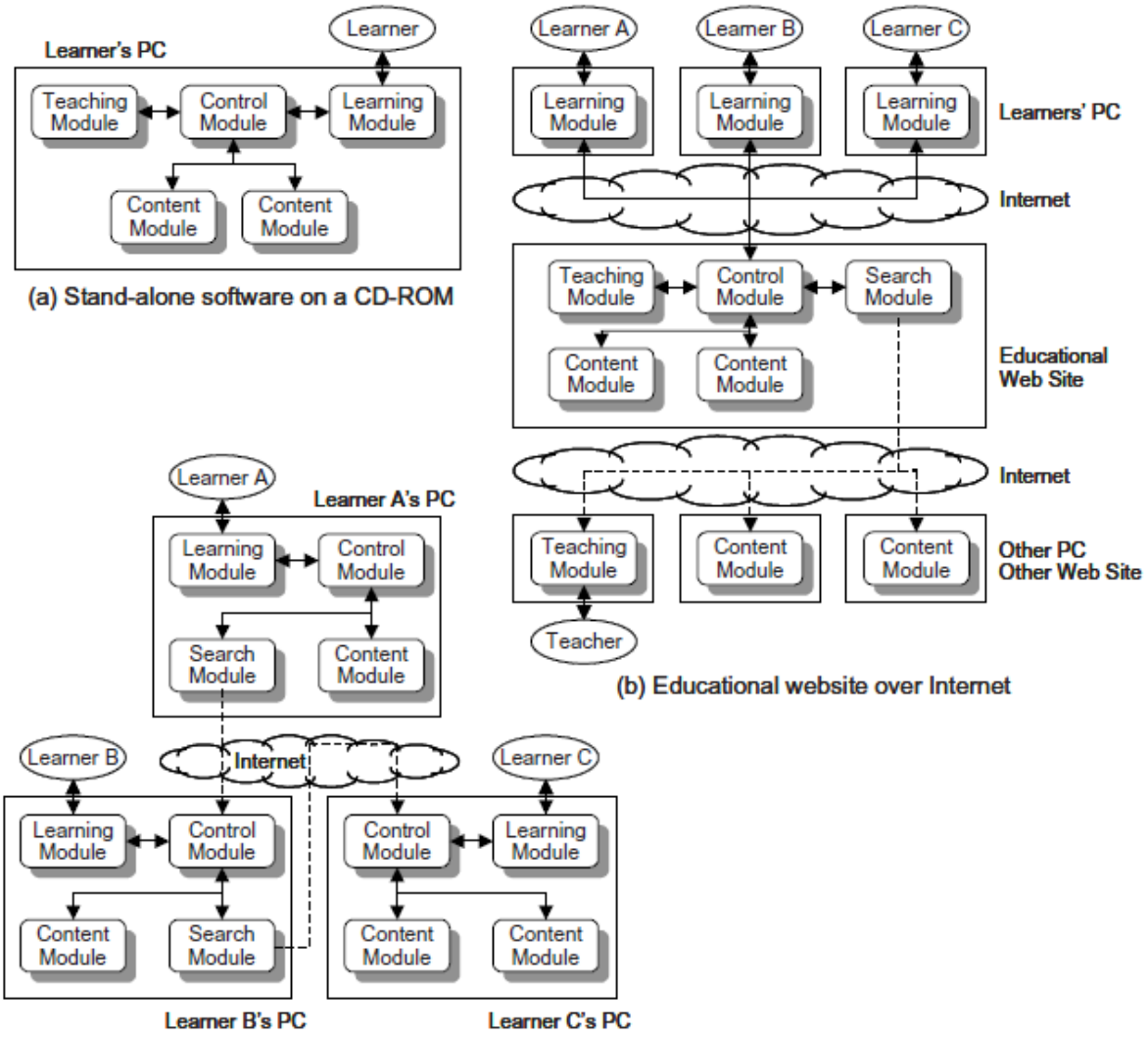
Yleisimmin kirjoittajat pohjasivat toteutuksensa joko suoraan IEEE:n standardiin tai sitä mukailevaan omaan moduulipohjaiseen arkkitehtuuriinsa. Esimerkkinä valmiista toteutuksesta on kuvassa 9 esitetty modulaarinen SCORM-pohjainen toteutus, joka nojaa LMS+CMS-pohjaiseen rakenteeseen.



Kuva 9. SCORM-pohjainen esimerkkiarkkitehtuurimalli. [Duan *et al.*, 2006.]

Kyseisessä mallissa on selkeästi erotettu kolme eri kokonaisuutta: käyttäjän selain, e-oppimisjärjestelmä ja eri toiminnallisuuksia toteuttavat moduulit (apparatus). Lisäksi kokonaisuuksien sisällä on eroteltu omiin moduuleihinsa mm. käyttöliittymä ja oppimisresurssit.

Duan *et al*. [2006] ottavat myös artikkelissaan kantaa moduulien väliseen kommunikointiin ja sen toteuttamisen mahdollisuuksiin. Tätä ei kuitenkaan tässä käydä sen tarkemmin läpi, koska se on varsin lähellä teknistä toteutusta, eikä siten ole tutkimuksen piirissä. Modulaarisuuteen keskittyvien kirjoittajien osalta selkeästi modulaarisin rakenne on Chon ja Leen [2004] artikkelissa "A Module-Based Software Framework for E-learning over Internet Environment" ratkaisussa, jossa koko toteutus pohjaa erilaisiin moduuleihin, joiden välinen kommunikaatio on toteutettu esimerkiksi internetin avulla. Kyseinen artikkeli ehdottaa toteutusta nimeltä "Modular Learning Engines with Active Search", joka pohjaa IEEE:n [2003] LTSA-arkkitehtuuriin ja tuo siihen päälle modulaarisuutta sekä LOM-tyylistä resurssien uudelleenkäytettävyyttä. Esimerkkitoteutus on kuvattu kuvassa 10.



Kuva 10. Modular Learning Engine. [Cho ja Lee, 2004.]

Toteutus pohjaa viiteen eri moduuliin (moduulien sisältöön ei erikseen otettu kantaa artikkelissa):

1. Oppimismoduuli
2. Opetusmoduuli
3. Sisältömoduuli
4. Hakumoduuli
5. Hallintamoduuli

Kaikilla moduuleilla on LTSA:n mukainen rajapinta, joten mikä tahansa moduuli voi keskustella minkä tahansa moduulin kanssa suoraan. Esitetyssä tapauksessa kommunikointi tapahtuu internetin

yli, mutta välitysmediaa ei ole toteutuksessa rajattu. Osa moduuleista sijaitsee käyttäjän työasemalla, osa verkkosivuilla ja osa internetissä palveluntarjoajilla. Koska yhteysrajapintaa ei ole rajattu, moduulien sijoittelu on toteutuksesta riippuen mahdollista käytännössä aivan rajoittamattomasti.

Käytännössä toteutus on P2P-tyylinen, jossa mikä tahansa moduuli voi olla yhteydessä mihin tahansa moduulin, ja yhden moduulin poistuminen ei vaikuta kokonaisuuden toimintaan. Artikkeleissa kuvattuja hyötyjä tällaiselle toteutukselle oli kuvattu seuraavia:

- Joustavuus mahdollistaa monenlaisia eri toteutustapoja ja -tyylejä
- Modulaarisuus mahdollistaa erikokoisia implementaatioita
- Standardien mukaisuus mahdollistaa laajennettavuuden ja integraation muihin järjestelmiin

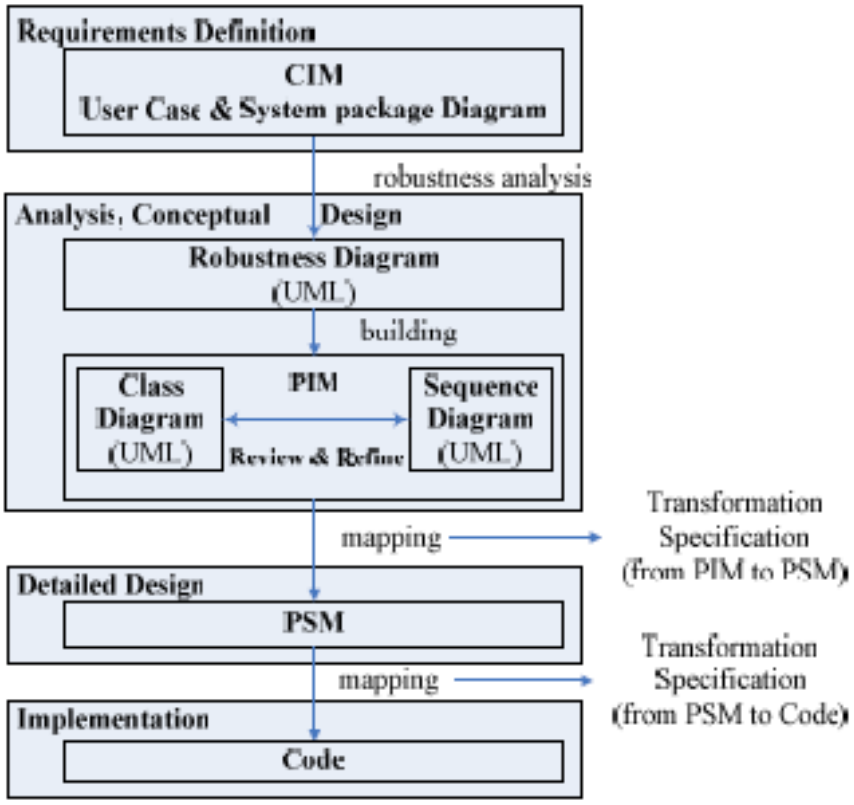
Toinen modulaarisuutta korostava teos on Congin *et al.* [2010] artikkeli aiheesta ”A Model-Driven Architecture Approach for Developing E-learning Platform”. Artikkelin pohjana on ajatus siitä, että nykyisissä e-oppimisratkaisuissa on ongelmia tiedon uudelleenkäytössä ja rakenteiden kertakäyttöisyydessä. Esimerkiksi tietoja on hyvin vaikea siirtää yhdestä e-oppimisjärjestelmästä toiseen. Kirjoittajat toteavat myös, että ratkaisut ovat yleensä sidottuja tiukasti tiettyyn bisneslogiikkaan. Tästä syystä kirjoittajat ehdottavatkin toteutusta, jossa bisneslogiikka ja arkkitehtuuri erotetaan omiksi moduuleikseen, jolloin kumpikaan ei sido toista. Näin mahdollistetaan järjestelmän monikäyttöisyys.

Muiksi ongelmiksi nykyjärjestelmissä Cong *et al.* [2010] listaavat ainakin:

- Oppimisobjektien uudelleenkäytön hankaluus (RLO)
- Sovelluskehyksen uudelleenkäyttöasteen vähyys
- Vaikeudet suunnitelman ja koodin yhteensovittamiselle

Kuten myöhemmin esiteltävän artikkelin ”The Design of Software Architecture for E-learning Platforms” tarjoamassa ratkaisussa, myös Cong *et al.* [2010] ehdottavat lähtökohdaksi bisneslogiikan määrittelemistä ja erottamista toteutuksesta. Kirjoittajien mukaan kannattaa rakentaa ensiksi bisneslogiikka ja sovelluskehys, ja näiden päälle alustariippumaton arkkitehtuuri (Platform Independent Model).

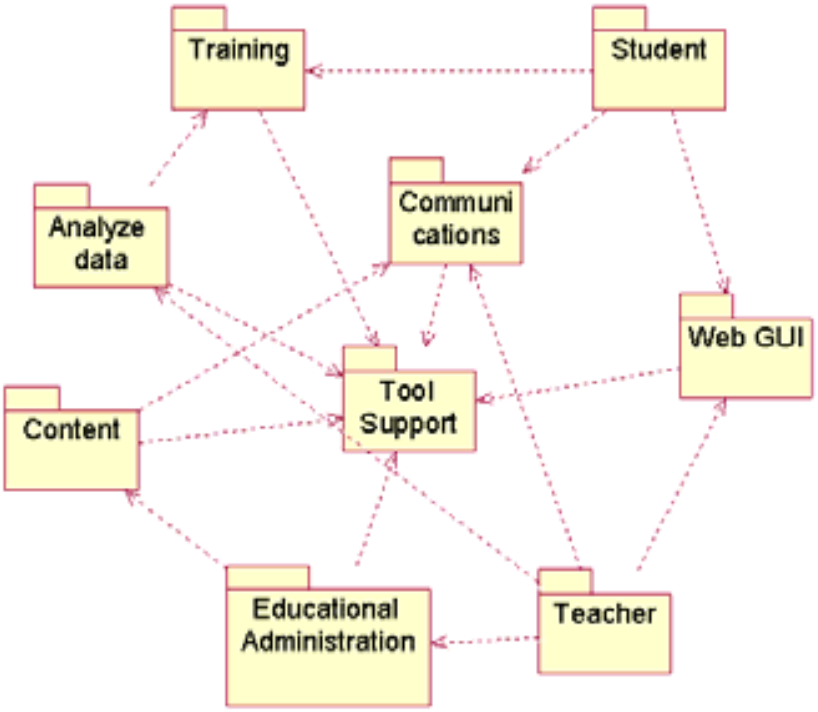
Congin *et al*. [2010] toteutus pohjaa Object Management Groupin (OMG) MDA-pohjaiseen ratkaisuun (Model-Driven Architecture Approach). MDA:n tarkoituksena on erottaa toteutusmalli ja toteutusimplementaatio toisistaan. Tätä kautta on päästy parempiin yhteensopivuuksiin, uudelleenkäytettävyyteen ja toteutuksien yhdenmukaisuuksiin. Congin *et al*. [2010] esittämä toteutus on esitetty kuvassa 11.



Kuva 11. Model-Driven Architecture Approach. [Cong *et al.*, 2010.]

Cong *et al.* [2010] viittaavat artikkelissaan Bizonovan *et al.* [2007] artikkeliin MDA-pohjaisesta toteutuksesta, jossa on kuvattu yleisellä tasolla ajatus e-oppimisjärjestelmien arkkitehtuurista. Tämä kuvaus pohjaa suosituimpien e-oppimisjärjestelmien, kuten Moodle ja OLAT, käyttämään arkkitehtuuriin. Cong *et al.* [2010] ottavat kuitenkin kantaa tässä mallissa olevaan ongelmaan; se ei erottele tarpeeksi selkeästi bisneslogiikkaa arkkitehtuurista. Sen vuoksi kaikki tähän malliin pohjaavat implementaatiot vastaavat vain tietynlaiseen bisneslogiikkaan, joten oli selkeä tarve kehittää mallista parempi versio, joka olisi vielä monikäyttöisempi.

Lähtökohtaisesti Congin *et al*. [2010] malli kuitenkin jatkaa Bizonova *et al.* [2007] työtä. Kuvan 11 malliin pohjaten Cong *et al.* [2010] jakavat prosessin erilaisiin valmiisiin kokonaisuuksiin ja jalostavat ylhäältä lähtien vaatimuksia ja bisnestarpeita aina valmiiksi toteutukseksi (koodiksi) asti. He lähtevät liikkeelle järjestelmäriippumattomasta mallista (Computation Independent Model (CIM), jossa määritellään käyttötapaukset ja järjestelmän ominaisuudet. Kuvassa 12 on esimerkkejä erilaisista ominaisuuksista, joita järjestelmästä voi löytyä.

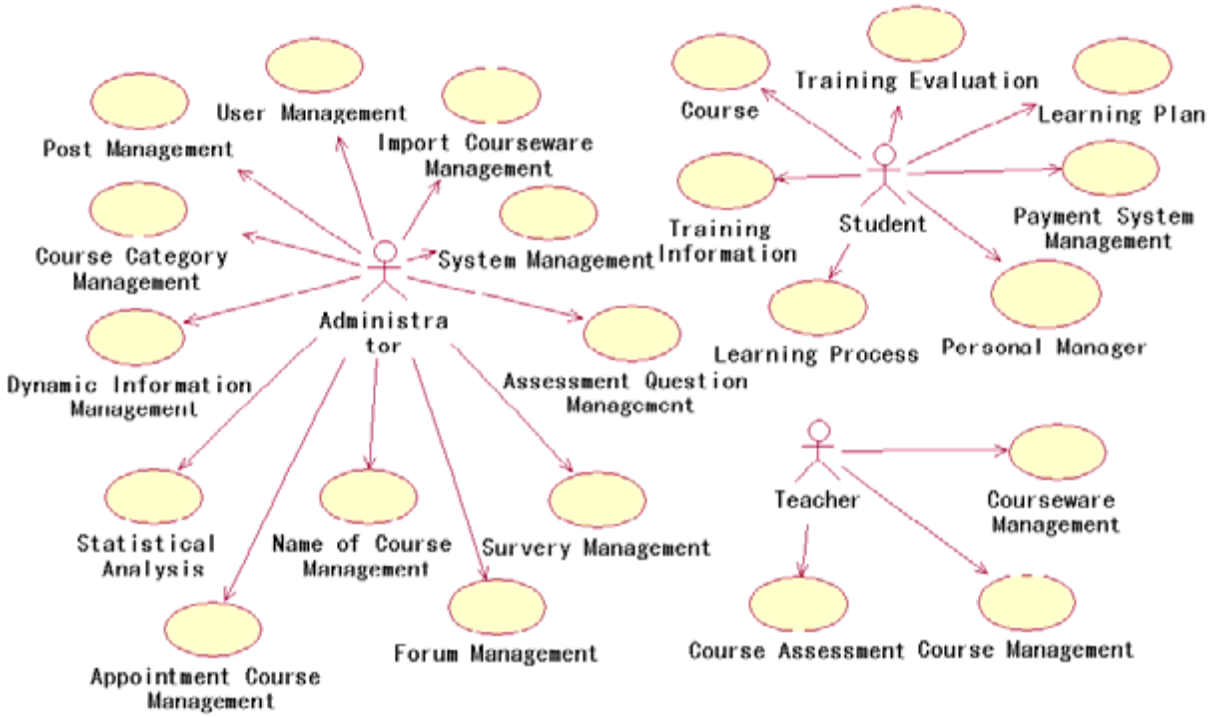


Kuva 12. Järjestelmän ominaisuuksia. [Cong *et al.*, 2010.]

Tarkoitus on siis kuvata korkealla tasolla erilaisia järjestelmään sekä sen toteutukseen ja toiminnallisuuksiin liittyviä kokonaisuuksia.

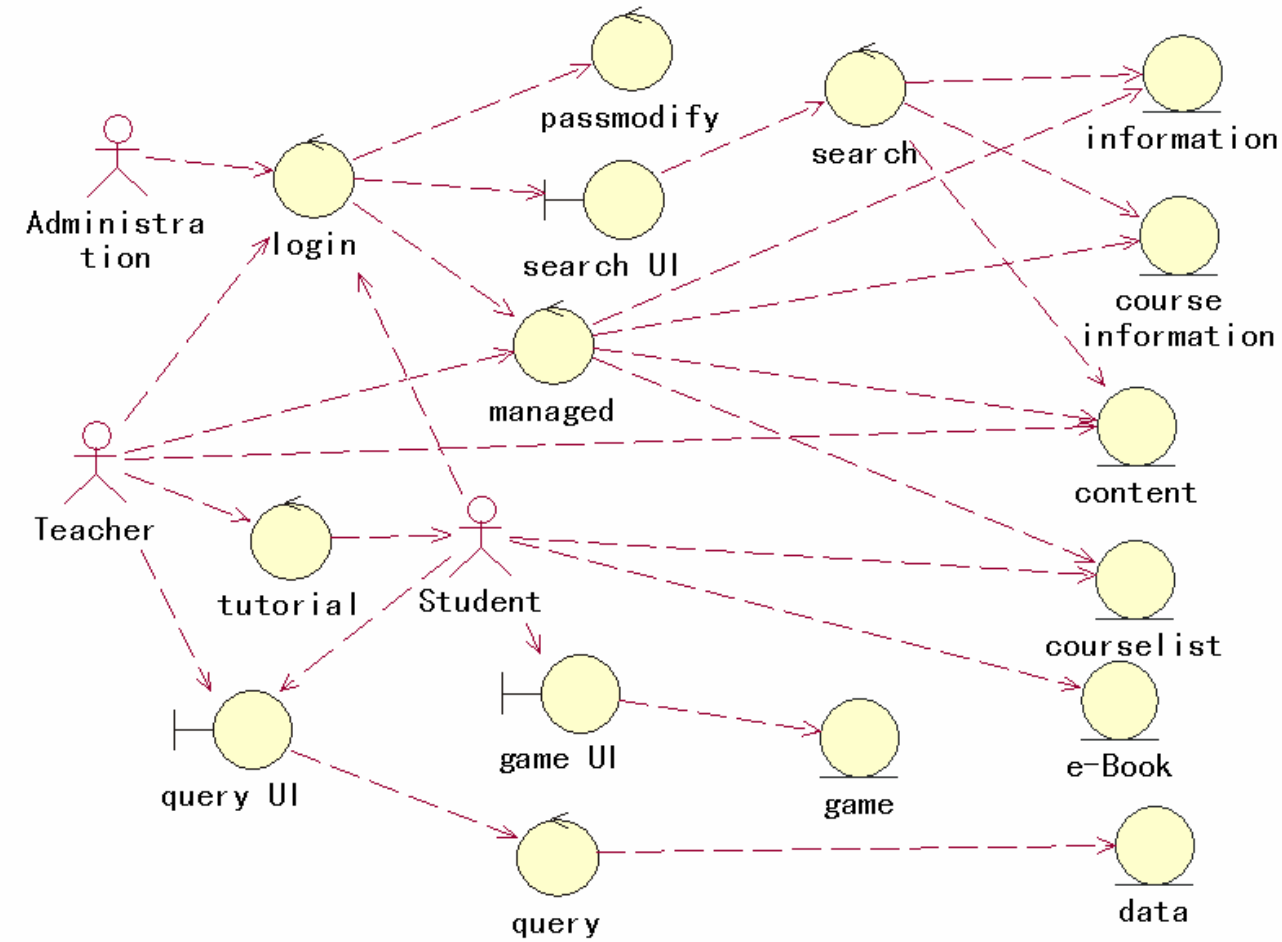
Edelleen kuvan 11 malliin pohjaten seuraavaksi mallinnetaan käyttötapauksia ja bisneslogiikkaa. Molempien pohjalla toimivat olemassa olevat standardit ja toteutukset, joihin voidaan nojata, kun määritellään bisneslogiikkaa tai haetaan esimerkkejä käyttötapauksille. Cong *et al.* [2010] nojaavat toteutuksissaan mm. LTSA:n ja SCORM:in malleihin.

Bisnesmallin lähtökohdaksi Cong *et al.* [2010] ottavat LMS:n (Learning Management System), jonka he määrittelevät olevan ydin bisnesmalli e-oppimisjärjestelmille. Tekniikaksi bisneslogiikan kuvaamiseksi he ehdottavat UML-kielellä (Universal Markup Language) toteutettua mallinnusta, jolla tulisi kuvata eri käyttötapaukset kuvan 13 mukaisesti.



Kuva 13. Käyttötapaukset. [Cong *et al.*, 2010.]

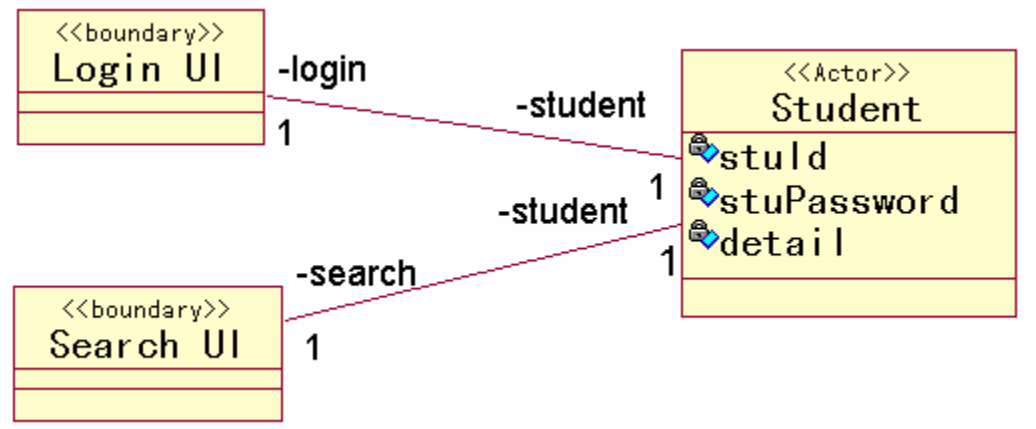
Jotta analyysi olisi luotettava, Cong *et al.* [2010] käyttävät mallissaan tekniikkaa nimeltä ”robustness analysis”. Tämän LEANin mukainen tekniikka mahdollistaa analyysi bisneslogiikan toteutumisesta käyttötapauksissa, ja varmistaa että käytetty terminologia on yhdenmukaista kautta eri mallien ja kuvausten. [Ambler, 2004.] Käytännössä kyseessä on UML-kielellä toteutettu kaavio, jossa on graafisin symbolein kuvattu käyttötapauksien toimijat, käyttöliittymät, prosessit, toimialueet ja käyttötapaukset kuvan 14 mukaisesti.



Kuva 14. Robustness analysis [Cong *et al.*, 2010.]

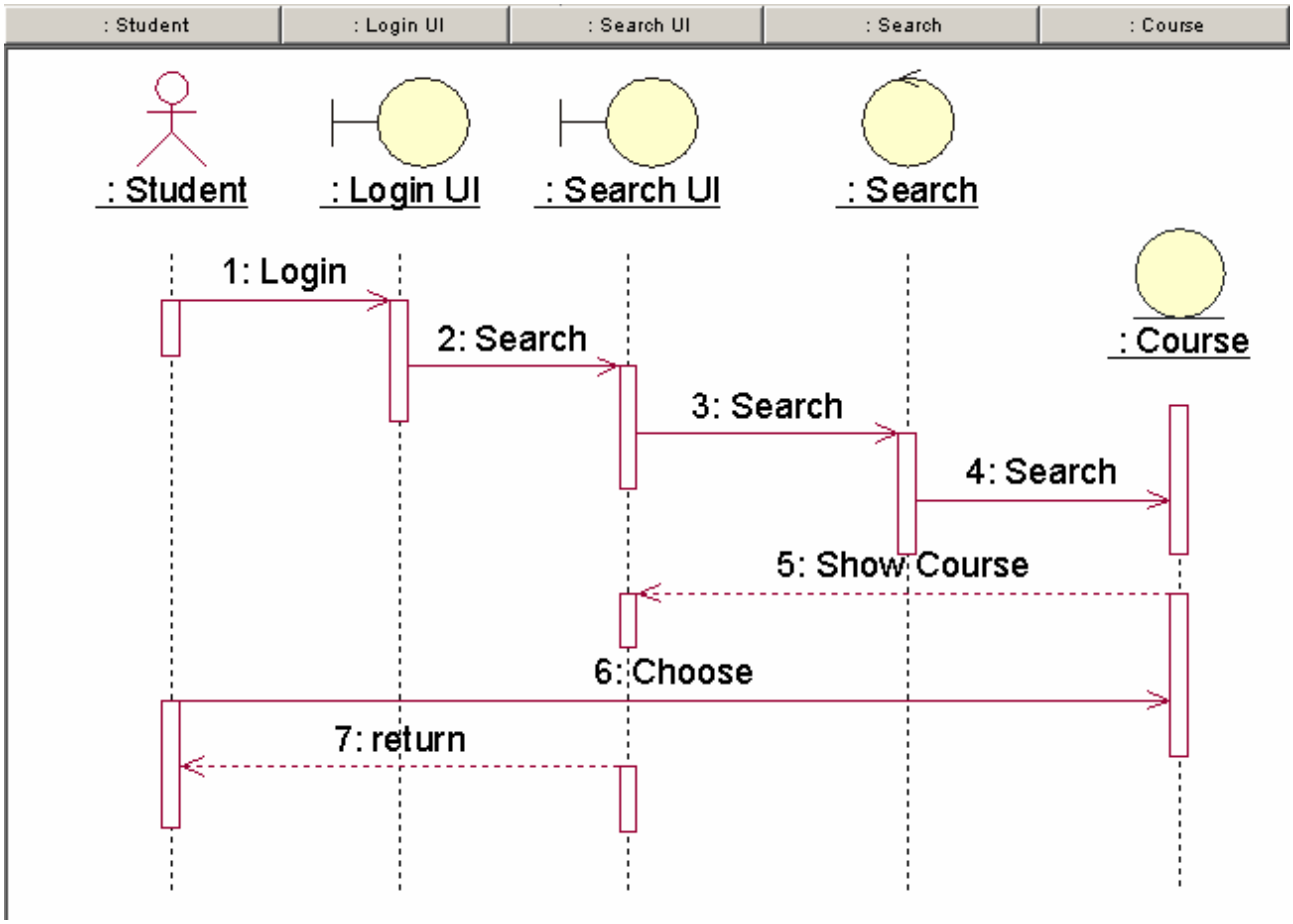
CIM-mallin pohjalta on tarkoitus jalostaa ylempien tason aineistoa tarkemmaksi ja tarkoituksenmukaisemmaksi aina alustariippumattomaan malliin asti (Platform Independent Model, PIM). PIM-mallissa yhdistetään lopulta ylhäältä tulleet järjestelmän ominaisuudet ja konseptuaaliset suunnitelmat käyttötapauksista sekä bisneslogiikasta.

Mallinnusta voidaan tämän jälkeen jatkaa luomalla luokkakaavio, johon artikkeli antaa hyvinkin tarkat ohjeet, mutta sen osalta sisältö ei ole tälle tutkimukselle relevanttia. Käytännössä luokkakaavio muodostetaan käymällä analyysin tulokset tarkemmalla tasolla läpi ja kuvaamalla ne luokkakaavioksi kuvan 15 tyyliin.



Kuva 15. Luokkakaavio. [Cong *et al.*, 2010.]

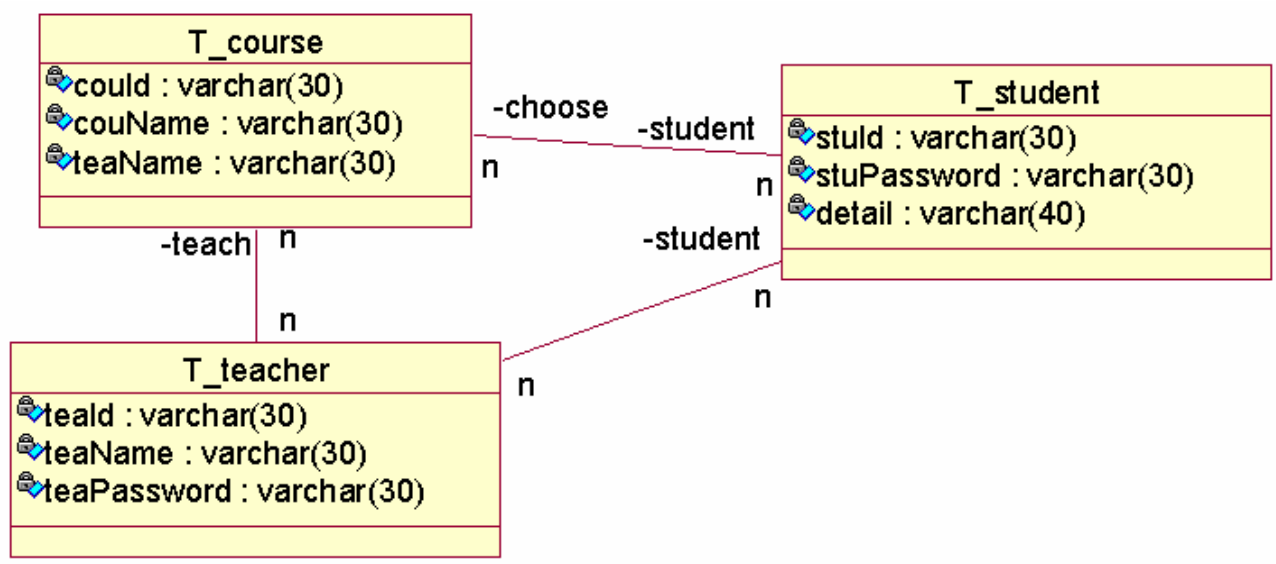
Tämän jälkeen mallissa luodaan vielä sekvenssikaaviot kuvan 12 analyysin pohjalta, mutta tämäkään ei ole enää relevanttia tutkimuksen kannalta. Käytännössä sekvenssikaavio luodaan käyttötapauksia ja analyysia hyödyntäen kuvan 16 kaltaisesti.



Kuva 16. Sekvenssikaavio. [Cong *et al.*, 2010.]

Tämän jälkeen on kuvan 11 mukaisesti jäljellä alustariippumattoman mallin (PIM) muuntaminen alustariippuvaksi malliksi (Platform Specific Model, PSM). Tässä kohtaa siirrytään hyvin spesifiselle tasolle toteutuksen mallinnuksessa. Osana mallinnusta Cong *et al.* [2010] valitsevat jo varsinaisen toteutusteknologiankin. Heidän valintansa on J2EE, mutta käytännössä (ja myös muissa malleissa osoitettuna) toteutuskieli voi olla mikä tahansa. Kielen tulisi tosin seurata kyseiselle kielelle määriteltyjä e-oppimisspesifisiä sovelluskehyksiä, mikäli sellaisia on määritelty.

Jokainen objekti ja tietue PIM-tasolla tulisi muuntaa varsinaista sovelluskehystä tukevaan muotoon PSM-tasolle. PSM-tasolla tarkoitetaan SQL-tyylistä tietojen mappausta ja tietokannan määrittelyä kuvan 17 tyyliin.



Kuva 17. PIM -> PSM -konversio. [Cong *et al.*, 2010.]

Viimeisenä vaiheena on vielä PSM-mallien muuntaminen varsinaiseksi ohjelmistokoodiksi. Tähän artikkeli ei kuitenkaan enää ota kantaa, vaan jättää varsinaisen implementaatiototeutuksen auki.

Yhteenvetona Cong *et al.* [2010] toteavat, että verrattuna klassisiin toteutuksiin heidän ajatuksensa mallien muuntamisesta ylimmältä abstraktilta tasolta alaspäin yksityiskohtaisemmaksi ja spesifisemmäksi aina varsinaiseen toteutuskieleen asti tarjoaa paljon paremman mahdollisuuden käyttää tasoja ja tehtyjä kaavioita sekä malleja uudestaan, sekä luoda mallin pohjalta tehokkaammin uusia järjestelmiä.

5.4. Muut mallit

Muiden mallien osalta ajatuksena oli tuoda yhteen asioita, jotka eivät nimellisesti kategorisoidu yhden otsikon alle, mutta joissa on selkeitä yhtäläisyyksiä, joiden vuoksi ne tulisi käsitellä kokonaisuutena.

Artikkelissaan ”The Design of Software Architecture for E-learning Platforms”, Zhou *et al* [2010] kuvaavat oman ajatuksensa e-oppimisarkkitehtuurista (kuva 18). Kyseinen arkkitehtuuri pohjaa kahteen pääkohtaan:

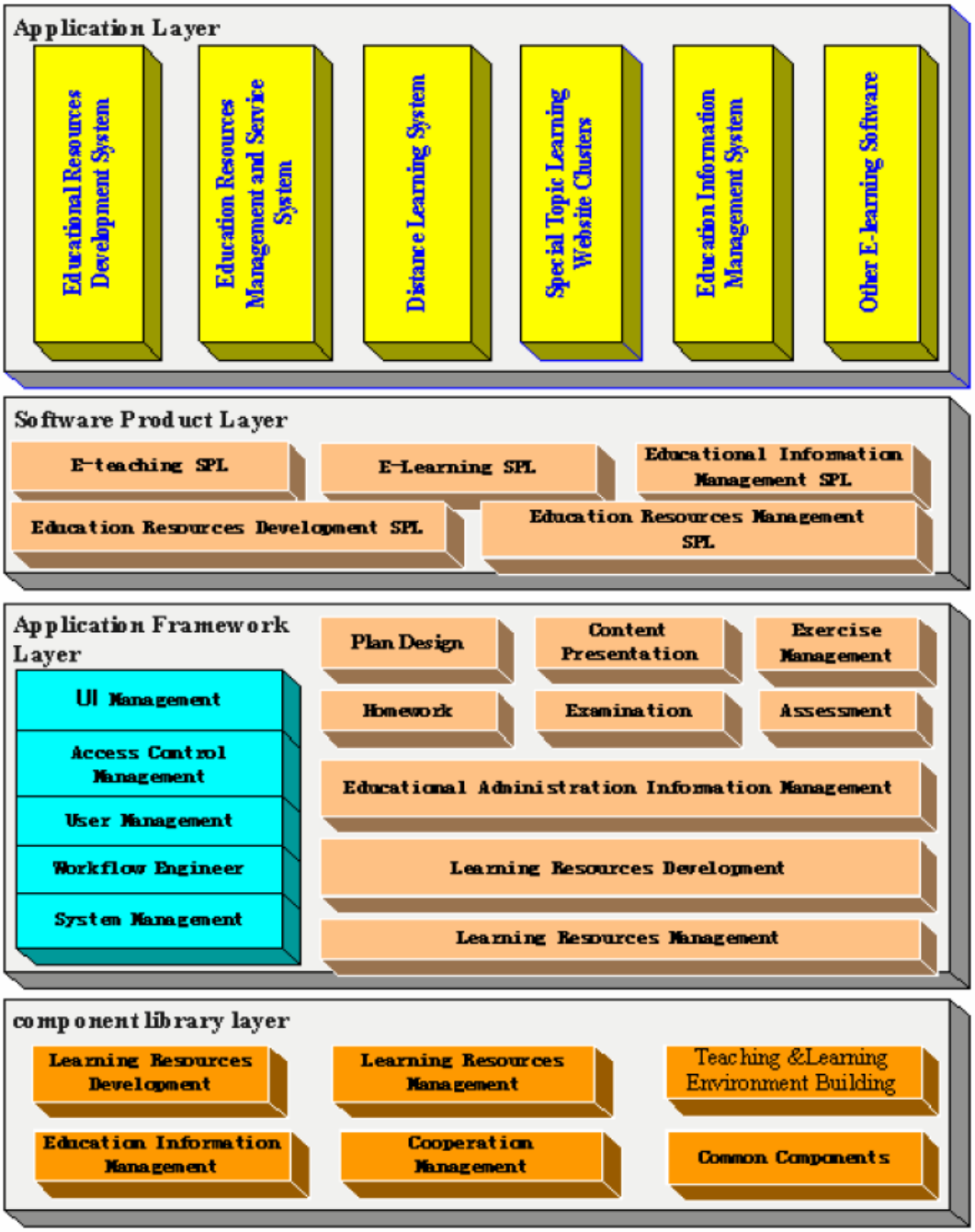
- Sovellusaluekohtainen ohjelmistoarkkitehtuuri (DSA)
- Ohjelmistojen tuotelinjat (SPL)

Tässä lähestymistavassa sisällytetään bisneslogiikka ohjelmistokomponentteihin, ja tämän päälle rakennetaan varsinainen e-oppimistoteutus. Käytännössä ensiksi rakennetaan komponentit, joiden pohjalta rakennetaan erilaiset tuotelinjat eri käyttötarkoitukseen, ja näistä muodostuu kokonaisuus, joka on e-oppimisjärjestelmä.

Zhoun *et al.* [2010] käyttämä toteutustapa eroaa oleellisesti aiemmin esitellyistä malleista, jotka pohjautuvat olio-malleihin tai komponenttipohjaisiin kehyksiin. Zhoun *et al.* [2010] mukaan sekä olio- että komponenttimalleissa on selkeitä ongelmia mm. pakollisissa periytymisissä ja uudelleenkäytettävyydessä

Varsinainen arkkitehtuuri jakautuu neljälle eri tasolle:

1. Sovellus (Application Layer)
2. Tuotantolinja (Software Product Layer)
3. Sovelluskehys (Application Framework Layer)
4. Komponenttikirjasto (Component Library Layer)



Kuva 18. E-oppimisarkkitehtuuri. [Zhou *et al*, 2010.]

Sovellustaso pitää sisällään valmiita e-oppimistoteutuksia, jotka on koostettu tuotantolinjan valmiista komponenteista. Ajatuksena on tarjota valmis implementaatio suoraan käyttöönotettavaksi valmiilla ominaisuuksilla. Tällöin käyttöönotto on helpompaa ja nopeampaa ja jokainen instanssi on identtinen.

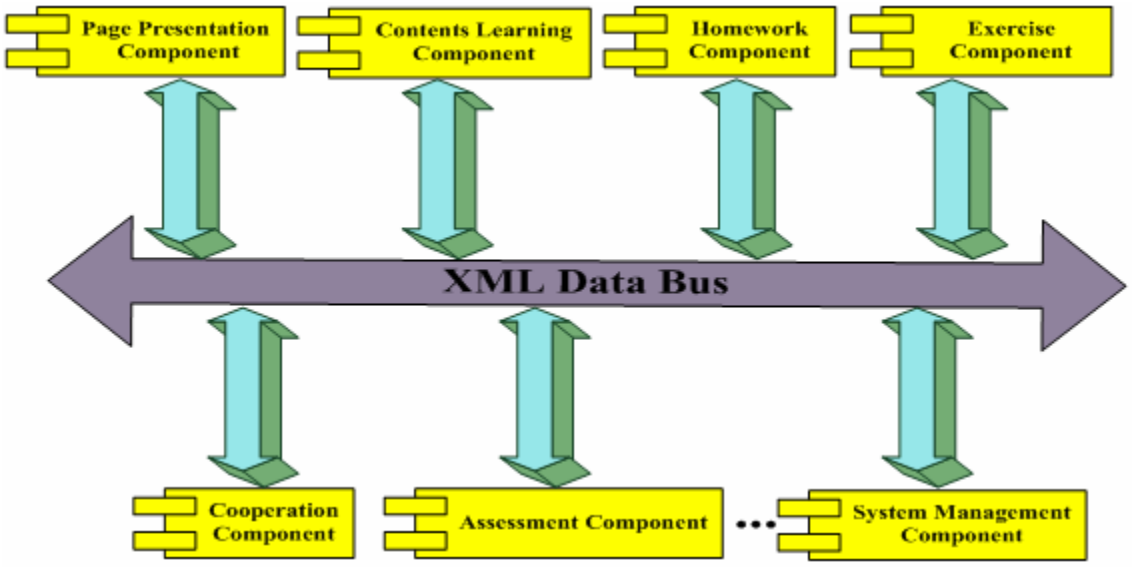
Erilaisia osakokonaisuuksia on tarkoitus hyödyntää riippuen siitä, millaista järjestelmää ollaan toteuttamassa, Näin voidaan säästää aikaa, kun ei tarvitse erikseen hakea ulkopuolisia määritelmiä tai komponentteja.

Tuotantolinjataso pitää sisällään viisi erilaista valmista komponenttia, joita voidaan käyttää valinnaisesti ja hyödyntää uudelleen riippuen toteutusmallista. Jokainen komponentti on siis yksittäinen kokonaisuus, jonka tarkoitus on toteuttaa jokin spesifinen toiminnollisuus. Jokaisen komponentin sisällä on kuvattu funktionaaliset ja ei-funktionaaliset toiminnallisuudet sekä liitännät toisiin komponentteihin. Tuotantolinjataso on koko arkkitehtuurin ydintaso. Kun siinä valitaan oikeat tuotantokomponentit ja sovelluskehykset, voidaan rakentaa hyvin erilaisia e-oppimisjärjestelmiä hyvin erilaisiin tarkoituksiin hyvinkin joustavasti.

Sovelluskehystaso tarjoaa käyttöön erilaisia sovellusaluekohtaisia ja ei-sovellusaluekohtaisia valmiita sovelluskehyksiä ja komponentteja sekä valmiita kehyksiä e-oppimisjärjestelmän toteutukseen. Jokainen kehys on määritelty suhteessa muihin komponentteihin ja sisältää rajapinnat ja rajoitukset näiden yhteistoiminnasta.

Komponenttikirjastotaso sisältää varsinaiset ohjelmistoarkkitehtuurin liittyvät kirjastot. Se on joustava ja kattava lähde erilaisille e-oppimisjärjestelmiin liittyville resursseille. Tämä kirjasto kasvaa ajan myötä, ja sen tarkoituksena on sisältää kaikki jo aikaisemmin käytetyt kirjastot ja mahdolliset kolmansien osapuolten kirjastot, jotka sopivat määriteltyjen ehtojen puitteisiin. Ajatuksena on käyttää kirjastoa nopeuttamaan ohjelmistokomponenttien tuotantoa ja varmistamaan, että yleisellä tasolla komponenteilla on tietyt säännöt ja rajapinnat.

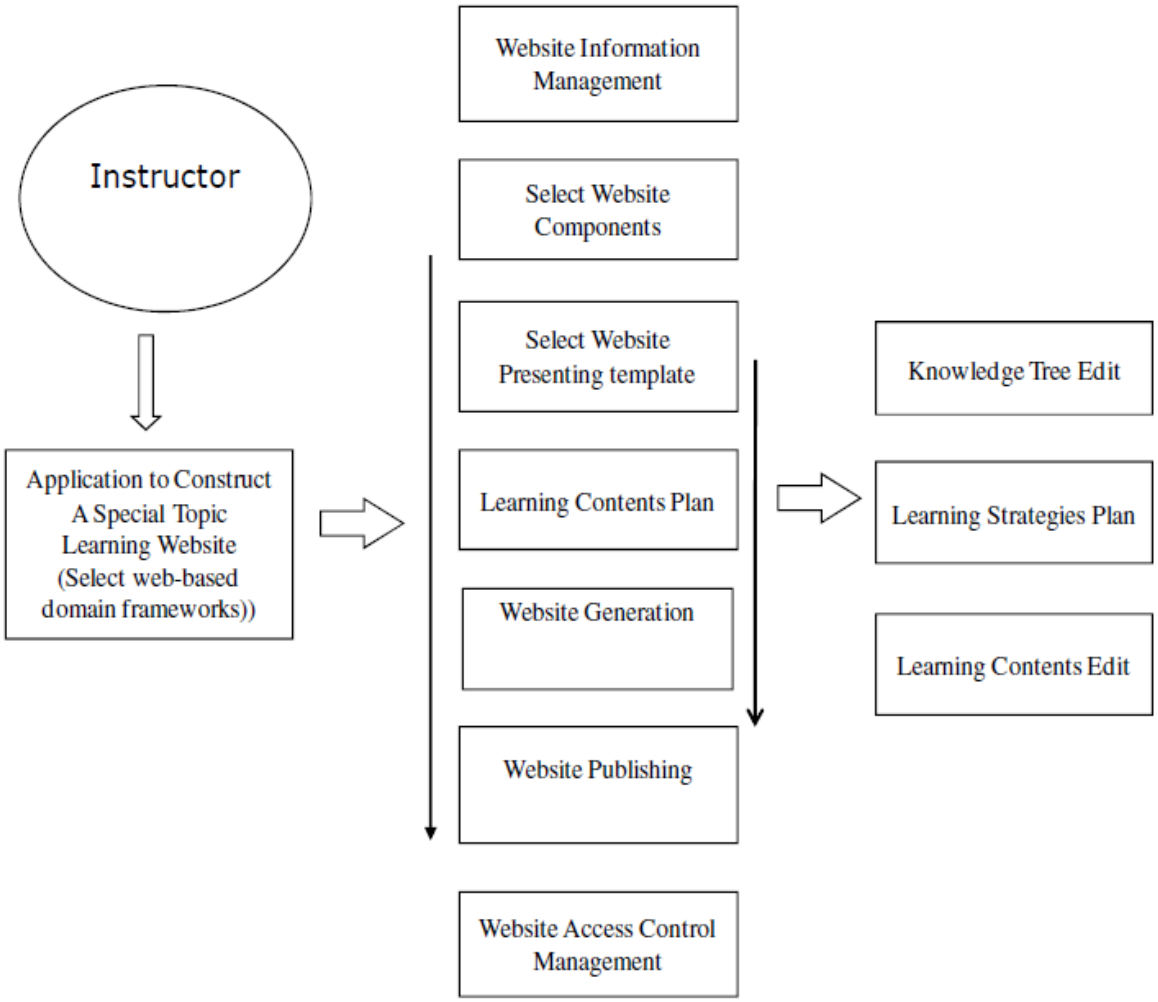
Zhoun *et al.* [2010] mukaan tasojen ulkopuolella voidaan hyödyntää standardeja UDDI/WSDL/SOAP-protokollia ja niiden avulla määritellä lähes mitä tahansa e-oppimisobjekteja uudelleenkäytettäviksi sovelluskomponenteiksi. Toteutusta varten Zhou *et al.* [2010] esittävät XML-väylämallia komponenttien koordinointia ja kommunikointia varten. Kuvattu XML-väylä on tietokanta itsenäisistä entiteeteistä. Koko väylä on siis XML-skeemalla kuvattu kokoelma tietueita (kuva 19).



Kuva 19. XML-väylä. [Zhou *et al.*, 2010.]

XML-väylän avulla kukin komponentti voi kirjoittaa ja lukea XML-dataa suoraan väylään toteuttaakseen omaa toiminnallisuuttaan ilman, että sen tarvitsee olla tietoinen muista komponenteista tai kommunikoida suoraan niille. Tiivistettynä XML-väylä toimii tiedon välittäjänä ja kanavana yksittäisille komponenteille kommunikointia varten. Näin yksittäiset komponentit eivät ole sidottuja muihin komponentteihin, ja kokonaisuus pysyy helpommin hallittavana ja laajennettavana.

Esimerkkinä kirjoittajat kuvaavat oppimisjärjestelmän käyttöönoton kuvan 20 mallilla.



Kuva 20. E-oppimisjärjestelmän web-toteutus. [Zhou *et al.*, 2010.]

Käyttäen määrittelemäänsä arkkitehtuurimallia Zhou *et al.* [2010] voivat tarjota tavan ottaa käyttöön erillisiä web-pohjaisia erityisaihealueiden oppimisjärjestelmiä. Kyseisen järjestelmän hyötynä on, että opettajien (instructor) ei tarvitse osata ohjelmointia tai ymmärtää järjestelmäarkkitehtuuria. Käytännössä arkkitehtuuria käydään läpi taso kerrallaan ja valitaan ne komponentit ja kirjastot, jotka sopivat kyseiseen toteutukseen, ja lähdetään niiden säännöillä ja rajoituksilla rakentamaan järjestelmää. Tasojen myötä tulevat erilaiset säännöt ja määritykset mm. käytetyistä rajapinnoista ja tiedon esittämismuodoista (esim. XML).

Kaikki tietojenvaihto sidotaan XML-väylään eikä toteutuksessa siten tarvitse ottaa kantaa komponenttien välisiin yhteyksiin, vaan kokonaisuus voidaan muodostaa suoraan yksittäisistä komponenteista ilman tarvetta sitoa niitä yhteen. Kun kaikki tarvittavat komponentit on valittu, järjestelmä tarjoaa ne näkyville automaattisesti, minkä jälkeen järjestelmä on valmis käytettäväksi.

Hyötyinä tällaiselle arkkitehtuurimallille kirjoittajat esittävät mm. joustavuutta ja hierarkkisuutta sekä tehokkuutta uudelleenkäytettävien ja laajentuvien kirjastojen myötä. Lisäksi säännöt rajapinnoista ja

standardoiduista esittämismuodoista mahdollistavat laajennettavuuden ja integraatiot muihin järjestelmiin. Käyttöönoton helppous ja joustavuus myös henkilöille, jotka eivät ymmärrä teknologiaa, on myös yksi suuri etu tällaisessa arkkitehtuurissa.

Toisenlaista arkkitehtuuria esittelevät Fang ja Sing [2009] artikkelissaan ”Collaborative learning using service-oriented architecture: A framework design”. Kirjoittajat esittävät SOA-pohjaista (Service Oriented Architecture) ratkaisua e-oppimisjärjestelmän arkkitehtuuriksi. Heidän mukaansa SOA-pohjaisuus toisi mukanaan modulaarisuutta, ja valmis SOA-sovelluskehys yhdistettynä muihin valmiisiin kehyksiin voisi mahdollistaa spesifisten e-oppimisjärjestelmien pystytyksen helposti ja tehokkaasti.

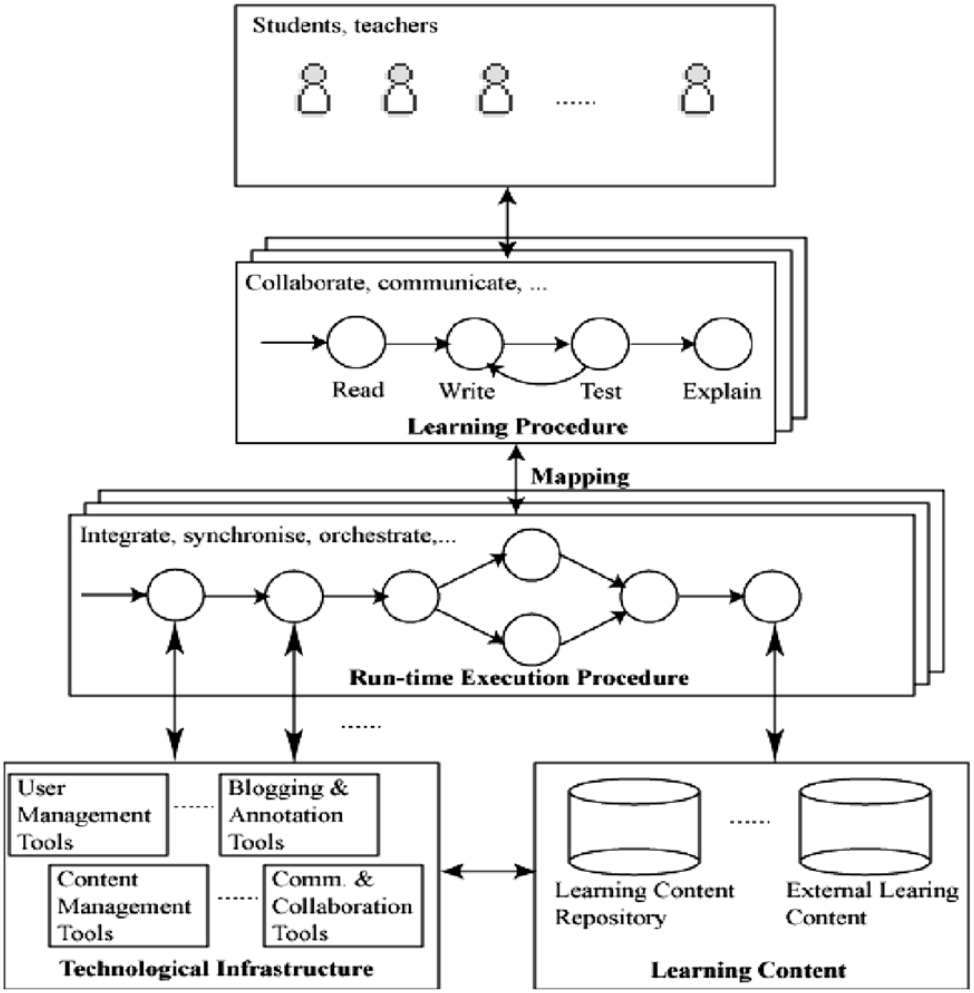
Kuten muidenkin kirjoittajien, myös Fangin ja Singin [2009] lähtökohtana on koodin, järjestelmän ja sen eri osien uudelleenkäytettävyys. He kokevat myös tärkeäksi erotella bisneslogiikka (eli käyttäjän toiminta) järjestelmän toiminnasta. Kirjoittajat erottelevat lähtötilanteeksi kaksi osiota, bisnesprosessit (BMP – Business Management Processes) ja SOA:n, ajatuksenaan hallita bisneslogiikan avulla myös oppimisprosesseja ja SOA:n avulla mahdollistaa parempi joustavuus eri järjestelmän komponenttien välillä.

Tarkennuksena SOA:n valintaan kirjoittajat esittävät, että vaikka nykyisin suurin osa e-oppimisjärjestelmistä on jo komponenteista muodostettuja, web-toteutus mahdollistaa SOA:n avulla vielä irtonaisempien komponenttien käytön ja niiden lisäämisen ja poistamisen tarpeen vaatiessa. Web-toteutus mahdollistaa myös paremman integraation muihin järjestelmiin ja helpomman ja tehokkaamman objektien uudelleenkäyttömahdollisuuden. Kirjoittajat käyvät läpi SOA:n hyödyt myös oppimisen kannalta, mutta ne eivät tämän tutkimuksen kannalta ole relevantteja. Päällimmäiseksi nousevat lähinnä hyödyt web-toteutuksen skaalattavuudesta, monimuotoisuudesta ja saavutettavuudesta.

Muiden kirjoittajien tapaan Fang ja Sing [2009] lähtevät rakentamaan kokonaisuutta bisnesprosessien kautta. Bisnesprosessi kuvaa ne tapahtumat, joissa aktiviteetit luovat käyttäjälle lisäarvoa muuntamalla syötteen joksikin arvokkaammaksi tuotokseksi.

Kirjoittajat mainitsevat, että yleensä e-oppimisessa käytetään joko sisältölähtöistä, työkalulähtöistä tai tehtävälähtöistä lähestymistapaa bisneslogiikkaan, mutta heidän mielestään mikään näistä ei ota tarpeeksi kantaa itse oppimiseen. Tämän vuoksi Fang ja Sing [2009] ehdottavat, että BMP:n tulisi sisältää sekä oppimiseen että järjestelmän käyttöön liittyvät toimintalogiikat. Käytännössä tämä tapahtuisi mallintamalla käyttötapaukset UML:n avulla vastaavasti kuin esimerkiksi Cong *et al.* [2010] kuvaavat aiemmin esitellyssä artikkelissaan ”A Model-Driven Architecture Approach for Developing E-learning Platform” (kuva 13).

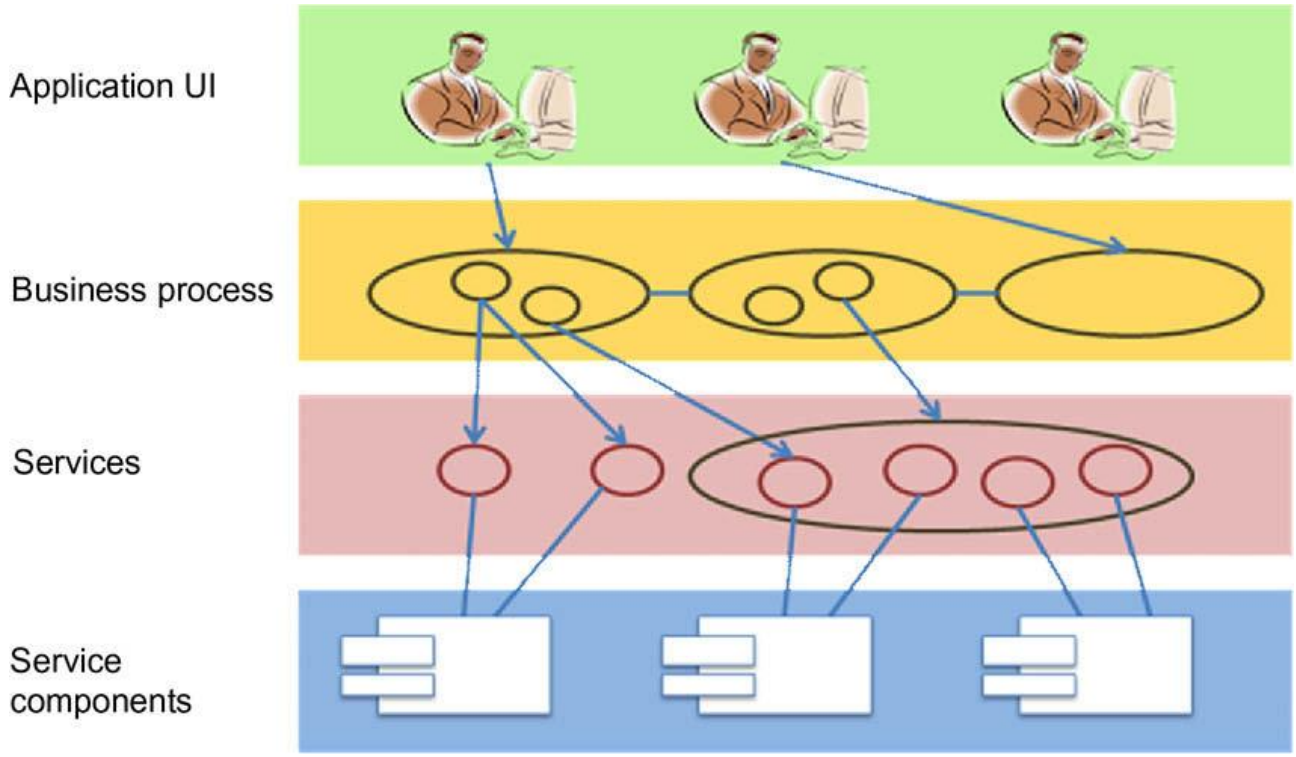
Tästä kirjoittajat jatkavat määrittelemällä, että kaiken toimintalogiikan on oltava sidottu palveluihin (services), jotta oppimisobjektit olisivat varmasti käytettävissä kattavasti web-toteutuksissa. Tämän jälkeen mitä tahansa palvelua pitää voida käyttää minkä tahansa sovelluksen tai sen osan toimesta.



Kuva 21. Geneerinen oppimisprosessi e-oppimisjärjestelmässä. [Fang ja Sing, 2009.]

Esimerkkinä palvelusta kirjoittajat kuvaavat oppilastietokannan, joka tarjoaa palveluina oppilaiden ilmoittautumis- ja rekisteröintiprosessit, joita voidaan sen jälkeen hyödyntää eri sovelluksien implementaatioissa (kuva 21).

SOA-pohjainen ratkaisu on siis kokoelma palveluita, joita hyödyntämällä voidaan koostaa toteutus. Toteutuksessa voi olla vain ne palvelut, joita tarvitaan sen hetkiseen toteutukseen, koska sitä on mahdollista mukauttaa myöhemmin poistamalla tai lisäämällä palveluita. Erityinen fokus toteutuksessa on palveluiden mallintamisessa, kuvaamisessa ja toteuttamisessa. Tämän SOA-mallin perustana kirjoittajat käyttävät neljän tason mallia (kuva 22).



Kuva 22. SOA-malli. [Fang ja Sing, 2009.]

Mallissa lähdetään liikkeelle oppimisprosesseista sekä bisnesprosesseista UI-tasolla. Nämä prosessit tulisi tunnistaa ja kuvata sellaisella tasolla, että niistä voidaan erotella omiksi palveluikseen selkeät kokonaisuudet. Toisin sanoen palvelut tulisi eritellä sellaisiksi kokonaisuuksiksi, että niitä voidaan käyttää uudelleen ja eri tilanteissa, eli tehdä niistä geneerisiä eikä spesifisiä.

UI-taso kuvaa käyttäjänäkökulmaa eli niitä toimijoita, jotka palveluita käyttäen toteuttavat bisnesprosesseja. UI-taso ottaa kantaa myös siihen ohjelmistoon, verkkosivuun, tai muuhun palveluun, jonka kautta e-oppimisjärjestelmän palveluja käytettään. UI-tasolla on siis tarkoituksena kuvata erilaisia käyttäjiä, tarpeita ja tavoitteita, ja tätä kautta muodostaa juuri heille sopiva implementaatio järjestelmästä. Käyttötapaukset, sellaisina kuin Cong *et al.* [2010] ne kuvaavat (kuva 13), voivat esimerkiksi toimia hyvänä lähtökohtana bisneslogiikan määrittämiselle.

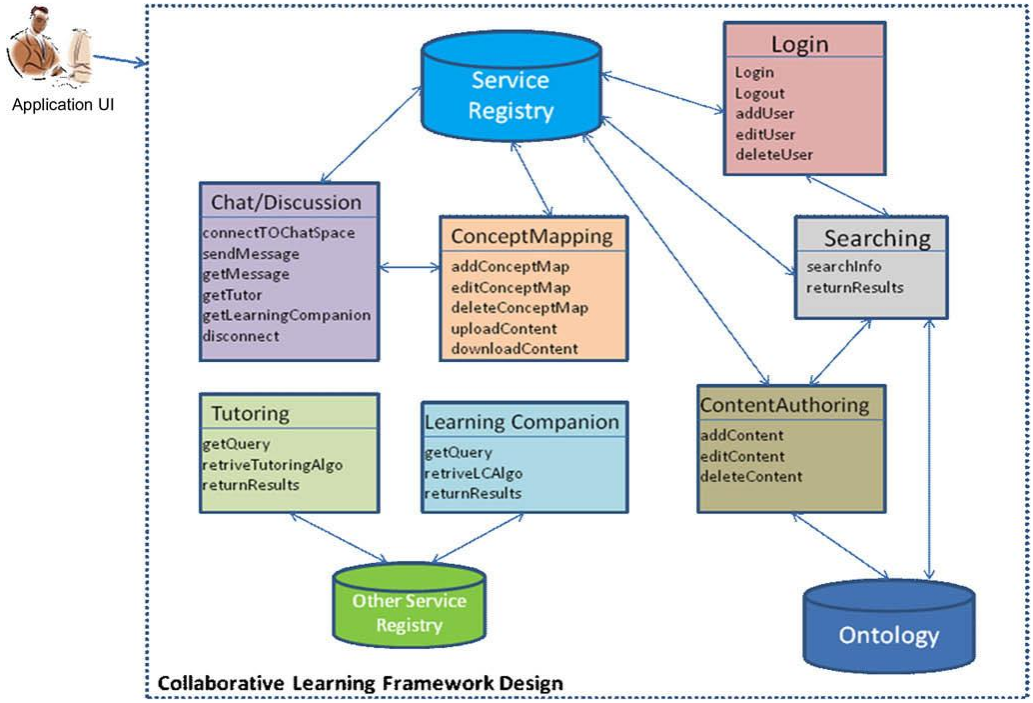
Bisnesprosessitasolla määritellään aktiviteetteja, jotka toteuttavat käyttäjien bisneslogiikkaa. Tarkoituksena on kuvata prosesseja ja/tai tehtävävirtoja, joissa monien eri tehtävien tai aktiviteettien kautta päädytään toteuttamaan jokin tarve. Esimerkkeinä kirjoittajat esittävät tapauksen, jossa oppija etsii informaatiota tietokannasta ja keskustelee oppimistehtävistä vertaistensa kanssa. Tällainen tapaus on mahdollista kuvata UML-kielellä esimerkiksi kuvan 12 tyylisesti. Bisnesprosessit siis tarkentavat UI-tason kuvausta ottamalla kantaa prosesseihin, ei pelkästään tarpeisiin tai aktiviteetteihin.

Tämän jälkeen seuraa palvelutaso. Fang ja Sing [2009] kuvaavat yleisimmät palvelut, joita SOA-pohjaisessa toteutuksessa on:

- Login
- Content Authoring
- Chat/Discussion
- Content Mapping
- Tutoring
- Learning Companion Services

Näiden palveluiden oletetaan yleensä olevan SOA-pohjaisten e-oppimisjärjestelmien rakenteessa. Nämä palvelut ja niistä tarjotut esimerkit kuvaavat sitä tasoa, jolla palvelut tulisi määritellä. Käymällä läpi bisneslogiikkaa ja -prosesseja ja katsomalla niitä käyttäjänäkökulmasta tulisi päätyä tämän tason palvelukomponentteihin.

Lopuksi SOA:n mukaisessa toteutuksessa pitäisi siis syntyä kuvan 23 mukainen palvelukonsepti, jonka pohjalta toteutusta voitaisiin lähteä rakentamaan olemassa olevien SOA-mallien ja standardien pohjalta.



Kuva 23. Palvelukonsepti. [Fang ja Sing, 2009.]

Artikkelissaan Fang ja Sing [2009] eivät vie SOA-mallia tätä pidemmälle, vaan kuvatussa ratkaisussa on tarkoitus ottaa kantaa vain siihen, miten palveluita tulisi miettiä ja löytää, ja tämän jälkeen varsinainen toteutus tulisi hoitaa aivan kuin missä tahansa muussa SOA-toteutuksessa. Fang ja Sing [2009] mainitsevat vielä lopuksi SOA-pohjaisen arkkitehtuurin tuovan erityisiä hyötyjä seuraavissa ominaisuuksissa:

- Joustavuus
- Skaalautuvuus
- Yhteensopivuus

Kirjoittajien näkemyksen mukaan perinteiset palvelin-sovellus-arkkitehtuurit jäävät näiden ominaisuuksien osalta vajaiksi, ja varsinkin komponenttien uudelleenkäyttö, saavutettavuus ja modulaarisuus ovat ongelmallista tällaisissa perinteisissä toteutuksissa.

5.5. Teknologiavalinnat

Osana malleja on kuvattu useita eri teknologioita, ohjelmointikieliä ja malleja. Koska ne eivät suoraan liity tutkimuskysymykseen, ne käydään läpi vain yleisellä tasolla siinä määrin kuin ne esiintyivät eri artikkeleissa.

SOAP: SOAP (Simple Object Access Protocol) on XML:ään (Extensible Markup Language) perustuva protokolla, joka huolehtii tiedonvälityksestä applikaatioiden ja käyttöjärjestelmien välillä. Se on siis viestintäprotokolla, joka mahdollistaa eri alustoilla (Windows, Linux) toimivien sovellusten kommunikaation alustariippumattomasti käyttäen http-protokollaa ja XML-formaattia. SOAP myös määrittelee XML-pohjaisten viestien rakenteen, jota web-pohjaiset sovellukset voivat käyttää viestinnässä. SOAP on siis samalla standardi, joka määrittelee, kuinka kommunikaatio tulisi toteuttaa XML:n avulla. [Rouse, 2014.]

UML: UML (Universal Markup Language) on OMG:n (Object Management Group) lanseeraama mallinnusmenetelmä, joka määrittelee eri tyylisiä diagrammeja, tarkoituksenaan standardoida tavat, joilla käyttäjät kuvaavat erilaisia asioita aina ohjelmistoista prosesseihin. [OMG, 2015.]

XML: XML eli Extensible Markup Language, on W3C:n määrittelemä joustava tekstiformaatti, jonka pohjana toimii SGML (ISO 8879). Alun perin sen tarkoituksena oli mahdollistaa elektroninen massajulkaisu; nykyään se on yleinen tiedonvälitysformaatti verkossa ja sen ulkopuolella. [W3C, 2015.]

J2EE: J2EE, eli Java 2 Platform, Enterpise Edition, on Javan alusta, joka on tarkoitettu erittäin suurille verkkoalustoille, joita tyypillisesti on käytössä suurissa yrityksissä. Se on suunniteltu erityisesti helpottamaan applikaatioiden kehittämistä kevyille asiakaspäätteille (thin client). Se määrittää

standardoituja, uudelleenkäytettäviä modulaarisia komponentteja ja mahdollistaa useilta osin automaattista ohjelmiston luontia. [Rouse, 2005.]

6. Yhteenveto

Finkin [2005] määritelmän mukaan yhteenvedon tarkoituksena on summata yhteen kirjallisuusselvitys. Se tulisi tehdä samalla huolellisuudella ja samoja linjauksia noudattaen kuin muukin selvitys, ja sen tulisi kyetä tuomaan esille löydöksiä, ja koota yhteen artikkelien sisältöä. Vastaavasti Okolin ja Schraham [2010] määrittelevät, että yhteenvedossa tulisi läpikäydä kirjallisuus ja sen kyky vastata tutkimuskysymykseen, esitellä uusia löydöksiä tai tuloksia, ja esittää mahdollisia jatkotutkimuksen aiheita. Varsinkin uusia ja odottamattomia löydöksiä pitäisi tuoda esille ja korostaa.

Näihin vaatimuksiin peilaten yhteenvedossa erotellaan erikseen materiaaliin liittyvät löydökset ja ajatukset, tutkimuskysymykseen vastaamisen, työn tulokset ja löydökset sekä jatkotutkimuksen aiheet.

6.1. Materiaali

Yleisesti ottaen työstä muodostui huomattavasti laajempi kuin alustava analyysi olisi antanut aihetta olettaa. Materiaalin suhteen oli odotettavissa, että löytyisi selkeitä, valmiita analyysejä erilaisista arkkitehtuureista nimenomaan e-oppimisympäristössä. Tähän pohjasi myös hakusanojen valinta sekä tietokantojen rajaukset. Näkemyksenä oli, että materiaalia löytyisi näiden rajausten avulla jo tarpeeksi, eikä kattavuutta tarvitsisi laajentaa. Näin ei kuitenkaan lopulta ollut; kukaan ei ollut vielä erikseen vertaillut arkkitehtuureja e-oppimisympäristöissä tai edes loppujen lopuksi tutkinut niitä kovinkaan paljoa, saati vertaillut eri malleja keskenään.

Yhtenä syynä tähän voi olla se, että e-oppiminen tieteenalana sekä teollisuudenalana on vielä hyvin nuori, eikä aihepiirissä ole tapahtunut sellaista kypsymistä, jonka pohjalta olisi päädytty selkeisiin vallitseviin teknologioihin tai arkkitehtuureihin. Vaikka hakutermejä vilisi artikkeleissa, varsinaiseen tutkimuskysymykseen liittyviä artikkeleita oli hyvin vähän. Mikäli aikaa olisi ollut enemmän, käytetyistä artikkeleista olisi voinut käydä läpi lähdeluettelon ja pyrkiä sitä kautta laajentamaan aineistoa ja saamaan mahdollisesti kattavamman analyysin.

Iteroimalla lähdemateriaalia enemmän olisi voitu päätyä laadukkaampiin tuloksiin, kuten Okolin ja Schabram [2010] mainitsevat:

> *”After completing one’s library and online review, it is important to supplement the search further to assure that all sources have been found and exhausted. Studying the reference section of landmark or particularly relevant pieces permits a ―backward search of other pertinent articles.”*

Vastaavasti myös Finkin [2005] mallissa todetaan, että laadukkaampiin tuloksiin päästäisiin iteroimalla aineistoa enemmän ja hakemalla lisäaineistoa enemmänkin artikkeleiden lähteiden kautta kuin suoraan uusia artikkeleita hakien. Näin voitaisiin päästä syvemmälle aiheen juuriin ja tätä kautta löytää selkeämpiä perusteluja artikkeleiden kirjoittajien tekemille valinnoille.

Kuitenkin työlle varatun rajoitetun aikaikkunan vuoksi kattava lähdemateriaalin iterointi oli työn valmistumisen kannalta mahdotonta. Koska työn laajuuden rinnalla on oleellista myös työn valmistuminen, Okolin ja Schabram [2010] ottavat kantaa tutkimuksen tarkkuuteen ja monimutkaisuuteen:

> *”Stand-alone literature reviews can and are conducted with varying standards of rigor, ranging from little more than an annotated bibliography to scientifically rigorous syntheses of a body of primary research.”*

Okolinin ja Schabramin [2010] mallissa tutkimus voidaan toteuttaa myös usean henkilön yhteistyönä, mikä suhteutettuna yllä mainittuun kommenttiin työn monimutkaisuudesta ja tämän työn aikatauluvaatimuksiin johti siihen, että tämän työn osalta varsinkin laajuutta jouduttiin rajoittamaan.

Muihin pro gradu -tasoisiin kirjallisuusselvityksiin verrattuna alkuperäisen materiaalin määrä ja laatu sekä lopullisen aineiston määrä olivat kuitenkin hyvin samalla tasolla, joten tässä mielessä voidaan nähdä, että selvitys oli aineiston kannalta onnistunut. Yhteensä aineistoa katselmoitiin otsikkotasolla 15 kappaletta, joista kuusi kelpuutettiin tarkempaan tutkimukseen. Näistä vielä yksi tippui pois tarkemmassa seulassa ja yhtä ei saatu ladattua, ja yksi lisättiin artikkelin sisällön pohjalta. Näin ollen kokonaan luettujen artikkeleiden määrä jäi viiteen kappaleeseen.

Lisänäkökulmana materiaalin osalta voitaisiin mainita, että hyvin moni artikkeli painottui teknisiin ratkaisuihin asti viemiseen, ja sitä kautta osa artikkeleiden sisällöstä rajautui suoraan pois tarkastelusta. Esimerkiksi Zhou *et al.* [2008] veivät ratkaisunsa valmiiksi tuotteeksi ja esittelivät sen ominaisuuksia kuvin ja tekstein, mutta koska tutkimuksen kohteena eivät olleet valmiit tuotteet, tällaiset asiat rajattiin pois käsittelystä.

Toisena parannuksena olisi nähtävissä lähteiden ristiinvertailu varsinkin, jos se olisi mahdollista tehdä automatisoidusti. Näin olisi mahdollista löytää artikkeleista yhteiset lähteet, joiden voisi olettaa olevan aiheen kannalta suuremmassa merkityksessä. Manuaalista lähteiden ristiinvertailua on melko työlästä tehdä, varsinkin jos aineistoa ja lähteitä on paljon.

6.2. Tutkimuskysymys

Alkuperäinen kappaleessa 3 esitetty tutkimuskysymys oli “Millaisia malleja e-oppimisarkkitehtuuriin löytyy kirjallisuuden perusteella?”. Kysymyksen laajuuden vuoksi siihen vastaaminen ei ole aivan

yksiselitteistä. Tästä syystä aineistoa pyrittiin jaottelemaan yhteisten löydösten kautta. Kuten kappaleessa 4 on eritelty, aineistosta löytyi seuraavia yhtenäisyyksiä:

- Selkeät standardit ja valmiit mallit
- Modulaariset ratkaisut
- Teknologiapohjaiset ratkaisut

Lisäksi malleissa yhdistyi usein eri tasoja, esimerkiksi SOA-malli perustui myös modulaarisuuteen, mutta lähestyi sitä omasta näkökulmastaan.

Kaikki löydökset ja havainnot osaltaan vastaavat tutkimuskysymykseen, ja vaikka selkeätä vertailua ei voitu tehdä tai päätellä, mikä malleista olisi paras, löydettiin useita yhtäläisyyksiä ja näkökulmia, jotka toistuivat artikkeleissa. Tutkimuskysymykseen vastaaminen tapahtuukin siis paloissa, ja jokainen niistä tuo oman elementtinsä kokonaisuuteen, josta vastaus muodostuu. Käytännössä ratkaisu muodostuu vastauksista ongelmiin, joita kirjoittajat olivat löytäneet tutkiessaan nykyisiä toteutuksia ja sitä kautta luodessaan omia arkkitehtuureitaan.

Selkeimmin artikkeleissa nousi esille tarve komponenttien uudelleenkäytettävyydelle; oli ilmeistä, että ainakin osassa nykyjärjestelmiä on suuria ongelmia sekä varsinaisen tiedon että järjestelmän osien uudelleenkäytettävyydessä.

> *”Most e-learning systems are not structured in a perfect architecture, and their elements are tightly coupled. So there are a lot of problems such as redundant development, difficulties to integrate with others and maintenance difficulties among e-learning software products.”* [Zhou *et al.*, 2008.]

Yhtenä ratkaisuna järjestelmien uudelleenkäytettävyydelle nähtiin mm. modulaarinen toteutus, kuten Dual *et al*. [2006] ja Cho ja Lee [2004] esittivät.

> “*The main objective is to combine OnlineLab researches and e-learning infrastructure in order to achieve the goal of increasing and enhancing learning opportunities and experiences for students.*” [Duan *et al.*, 2006.]

Modulaarisuuden etuina nähtiin mm. mahdollisuus rakentaa järjestelmää valmiista osista, joita pystyttäisiin hyödyntämään myöhemmin uusissa järjestelmissä ilman uudelleenrakentamista. Modulaarisuus mahdollistaa myös järjestelmän myöhemmän muuttamisen osia poistamalla ja lisäämällä. Lisäksi modulaarisuuden käyttäminen mahdollistaa järjestelmien rakentamisen myös sellaisten henkilöiden toimesta, joilla ei ole ohjelmointikokemusta.

> *”Most e-learning systems are not developed for customization. They cannot meet all requirements of instructors and learners in their teaching and learning activities.*

> *Meanwhile, most instructors of K-12 are not good at developing software, and cannot build a complicated software system on their own."* [Zhou *et al*., 2008.]

Fang ja Sing [2009] taas ottivat selkeästi kantaa nykyisten mallien palvelin-sovellus-arkkitehtuurin ongelmiin ja ehdottavat ratkaisuksi omaa SOA-pohjaista arkkitehtuuriaan.

> *"Many of the collaborative learning systems are mainly based on client-server architectures, which give rise to the problems of poor flexibility, scalability and interoperability. Collaborative learning using Service-oriented architecture, which serves as a novel approach in the e-learning domain, helps in distributing the learning content more efficiently and promotes reusability."* [Fang ja Sing, 2009.]

Myös bisneslogiikan tuominen tiiviisti mukaan järjestelmien suunnitteluun nousi esille lähes kaikissa artikkeleissa. Kun järjestelmiä tutkittiin taloudellisesta näkökulmasta, Cong *et al.* [2010] olivat huolissaan järjestelmien elinkaaresta ja varsinkin niiden käytettävyydestä tulevaisuudessa.

> *"At present, many teams have committed to the development of E-learning platform, but still there exist many problems in the process of development. First, it is difficult to use learning object from an LMS in another one and it can cause great economical problems in the future of these technologies. Second, the business logic and the platform technology were mixed in the process of system design."* [Cong *et al.*, 2010.]

Toinen bisneslogiikasta kumpuava selkeä elementti oli, että nykyisissä järjestelmissä käyttäjien tarpeita ei ollut otettu huomioon tarpeeksi selkeästi. Tämän vuoksi esimerkiksi Congin *et al.* [2010] artikkelissa lähdettiin liikkeelle heti aluksi käyttäjien tarpeista ja toimintalogiikoista, ja vasta tämän jälkeen keskityttiin järjestelmän muihin toiminnallisuuksiin.

Lähimmäksi selkeitä yhteisiä arkkitehtuureja nousivat mm. Duanin *et al.* [2006] sekä Chon ja Leen [2004] esittämät IEET:n standardit LTSA:sta, sekä Cong *et al.* [2010] mainitsema SCORM.

> *"This paper proposes Modular Learning Engines with Active Search (MOLEAS), an E-learning software framework over Internet environment. It overcomes some problems in Sect. 2 by employing information technologies. It has the following features: Standardized architecture based on IEEE P1484 Learning Technology Standard Architecture (LTSA)"* [Cho ja Lee, 2004.]

Näistä LTSA on IEEE:n standardi, joka määrittelee korkean tason arkkitehtuurin tietojärjestelmäavusteisille oppimis-, koulutus- ja opetusjärjestelmille ja kuvaa korkealla tasolla järjestelmän suunnittelun ja komponentit. SCORM on Yhdysvaltain puolustusministeriön määrittelemä kokoelma standardeja ja määrityksiä, joiden tarkoitus on kuvata web-pohjaisten elektronisten oppimisjärjestelmien toteutus. Edellä mainittuihin määritelmiin ja standardeihin

viitattiin lähes kaikissa artikkeleissa jollain tasolla ja niiden päälle tai niihin pohjautuen oli toteutettu suurin osa artikkeleissa mainituista toteutuksista.

Myös tulevaisuutta oli ajateltu monessa artikkelissa; esimerkiksi Fang ja Sing [2009] olivat huolissaan järjestelmien saavutettavuudesta ja integroinnista muihin järjestelmiin tulevaisuudessa. Tämän vuoksi komponenttien integraatioon otettiin kantaa suunnittelussa ja pyrittiin varmistamaan, että tieto olisi helposti siirrettävissä ja komponentit itsessään integroitavissa muihin järjestelmiin.

Myös Congin *et al.* [2010] mainitsema taloudellinen näkökulma on esimerkki tulevaisuuden huomioon ottamisesta. Tällaisen ajattelun pohjalla on tieto siitä, että järjestelmien rakentaminen on työlästä ja kallista, ja hyvän arkkitehtuurin avulla järjestelmien elinkaarta voidaan pidentää muokkaamalla niitä käytön aikana ja hyödyntämällä niiden osia uusissa järjestelmissä.

> *”[…] the business logic and the platform technology were mixed in the process of system design. Once the platform changed, it will not be able to reuse the model […]”* [Cong *et al*. 2010.]

Yhteenvetona voidaankin todeta, että e-oppimisarkkitehtuuriin ei löydy kirjallisuudesta ns. best-practice-mallia, vaan oikean arkkitehtuurimallin valinta on enemmän kiinni siitä, että ymmäretään, millaisia vaikuttavia seikkoja arkkitehtuureissa ja arkkitehtuurimalleissa tarvitsee huomioida, ja kuinka niiden avulla voidaan sekä arvioida olemassaolevia malleja että kehittää omia.

6.3. Työn tulokset ja löydökset

Okolin ja Schabram määrittelevät kirjallisuusselvityksen löydökset seuraavasti:

> *”Beyond simply reporting the procedures, the literature review should conclude by highlighting any novel findings. Was the literature supportive of particular existing theory, or did the reviewers establish a new model which builds on existing theory and will contribute to future research? Unexpected results, in particular, should be highlighted.”*

Tässä kappaleessa selvitetään tämän määritelmän pohjalta, miten löydökset tukevat olemassa olevia käytäntöjä, ja onko niiden pohjalta mahdollista määrittää uusia käytäntöjä ja/tai malleja.

Artikkeleissa viitattiin valmiiden mallien suhteen sekä LTSA:han että SCORMiin. Koska artikkelit lähes poikkeuksetta pohjasivat teoriansa jompaankumpaan näistä määrittelyistä, voidaan nähdä, että löydökset tukevat olemassa olevia käytäntöjä ja käytännössä keskittyvät laajentamaan ja/tai modernisoimaan niitä. Selkeiten viittaukset voidaan esittää taulukkomuodossa, johon on kasattu artikkeleiden sisältö ja oleellisimmat mallit sekä ominaisuudet (kuva 24).

Aineisto	Kirjoittaja	Pohjalla	Teknologiat	Erityisominaisuudet
An Architecture for Online Laboratory E-learning System	Duan *et al.* [2006]	SCORM (Valkoinen talo) LMCS (LMS + CMS [RLOs])	-	Tiedon käyttämisen (LMS) erottaminen sisällön hallinnasta (CMS)
A Module-Based Software Framework for E-learning over Internet Environment	Cho ja Lee [2004]	LTSA (IEEE) LOM (IEEE)	-	Modulaarisuus hyödyntäen LTSA:n mukaisia rajapintoja
A Model-Driven Architecture Approach for Developing E-learning Platform	Cong *et al.* [2010]	MDA (OMG) LTSA (IEEE) SCORM (Valkoinen talo) Moodle OLAT	UML Robustness Analysis CIM PIM J2EE	Bisneslogiikan erottaminen toteutuksesta Mallinnetut toteutukset eri käyttöskenaarioille
The Design of Software Architecture for E-learning Platforms	Zhou *et al.* [2010]	-	DSA SPL UDDI WSDL SOAP XML	Valmiisiin komponentteihin ja osakokonaisuuksiin pohjaava arkkitehtuuri Mahdollisimman paljon erottelua osien ja toimintojen välillä Bisneslogiikka irti toteutuksista Selkeä integroitavuus osien kesken
Collaborative learning using service-oriented architecture: A framework design	Fang ja Sing [2009]	SOA	UML	Bisneslogiikan erottaminen toteutuksesta Palveluihin perustuva rakenne Palveluiden välinen kommunikointi yhteisin menettelyin

Kuva 24. Arkkitehtuurien yhteenveto.

Selkeimpiä löydöksiä artikkeleista olivat jo aikaisemmin mainittu modulaarisuus, uudelleenkäytettävyys sekä elinkaariajattelu.

Modulaarisuuden edut ovat selkeät ja modulaarisuutta hyödynnettiin lähes jokaisen artikkelin lopullisessa toteutuksessa. Kattavaa vertailua sen suhteen, kuinka moni moderni järjestelmä oikeasti tukisi modulaarisuutta, ei ollut mahdollista tehdä tämän työn puitteissa, joten tällainen vertailu jää yhdeksi mahdolliseksi jatkotutkimuksen kohteeksi.

Uudelleenkäytettävyys etenkin Duanin *et al.* [2006] mainitsemia RLO:ita käyttäen nousi esille useaan otteeseen. Uudelleenkäytettävyyden lopullinen toteutustapa ei toisaalta ollut aivan yksiselitteistä. Eri malleissa käytettiin erilaisia tapoja, joiden avulla pyrittiin tuomaan järjestelmään uudelleenkäytettävyyttä. Esimerkiksi Zhou *et al.* [2008] pyrkivät luomaan valmiita ohjelmistokomponentteja, joiden avulla uudelleenkäytettävyyttä saataisiin aikaan. Cong *et al.* [2010] puolestaan pyrkivät erottelemaan bisneslogiikan kokonaan teknisestä toteutuksesta, jolloin järjestelmän osia voitaisiin paremmin hyödyntää erilaisissa tilanteissa, joissa bisneslogiikka voisi olla

erilainen. Tähän pohjaa esille tullut näkökulma elinkaariajattelusta ja käyttäjälähtöisyydestä sekä bisneslogiikan mukaan tuomisesta jo arkkitehtuurin määrittelyn alkuvaiheissa.

Kirjoittajista ainakin Fang ja Sing [2009], Zhou *et al.* [2008] sekä Cong *et al.* [2010] ottavat selkeästi kantaa bisneslogiikan määrittelyyn ja erotteluun. He pyrkivät tällä mahdollistamaan järjestelmän käytettävyyden hyvin erilaisissa tilanteissa ja mahdollisimman vähäisen tarpeen koodimuutoksille, mikäli järjestelmää muokataan eri tilanteisiin tai käyttötarkoituksiin.

Käyttäjälähtöisyyden osalta esimerkiksi Cong *et al.* [2010] vievät määrittelyn ja kuvaamisen erittäin tarkalle tasolle, jolloin ymmärretään tarkasti, millaista järjestelmää ollaan rakentamassa ja mihin tarkoitukseen. Kun käyttäjien tarpeet on paremmin ymmärretty ja kuvattu, järjestelmään ei kohdistu niin suuria muutospaineita käyttöönottovaiheessa, ja kun käyttäjätapaukset on selkeästi eroteltu osana ohjelmakoodia, on järjestelmän muokkaaminen ja uudelleenkäytettävyys paremmalla tasolla.

> *”At the same time, by the PIM modeling, the business model will be at a higher degree of reuse. It provides better solutions for the multiple applications of one design and the evolution of software.”* [Cong *et al*., 2010.]

Kokonaisuudessaan voidaan nähdä, että tutkimus oli onnistunut; siinä saavutettiin sille asetettuja tavoitteita riittävästi ja löydettiin tietoa ja yhteneväisyyksiä lähdemateriaalista. Näiden yhteneväisyyksien avulla voitiin koostaa selkeitä kokonaisuuksia, joihin pohjata e-oppimisjärjestelmien arkkitehtuurit. Varsinaista mittaria systemaattisen kirjallisuusselvityksen onnistumiselle ei kuitenkaan ole olemassa, vaan määrityksenä on enemmänkin se, että tutkimus on tuottanut uutta tietoa ja että se voidaan tarvittaessa toistaa käyttäen hyväksi selvityksessä kuvattuja menettelyjä. [Okolin ja Schabram, 2010.]

Yhteenveto löydöksistä voidaan esittää kuvan 25 taulukon avulla. Siihen on kerätty eri artikkelien kirjoittajien näkemykset aiheeseen liittyvistä ongelmista, niiden seurauksista ja esiteityistä ratkaisuista. Nämä ratkaisut voidaan nähdä pohjana uusille tutkimuksille tai valinnoille, kun mietitään arkkitehtuurimallia e-oppimisratkaisulle.

Ongelma	Seuraus	Ratkaisu	Kirjoittaja
Komponenttien uudelleenkäytettävyyden puuttuminen	Tietoa ei voida helposti siirtää järjestelmien välillä Samaa tietoa voidaan joutua syöttämään useita kertoja Integraatio muihin järjestelmiin vaikeaa Järjestelmien päivitys vaikeaa Järjestelmien rakennetta vaikea muuttaa jälkeenpäin	"Learning Object Metadata"-standardin hyödyntämien tietojen kuvaamisessa	Duan *et al.* [2006]
		RLO-tyylinen objektien määrittely	Duan *et al.* [2006]
		Järjestelmän eri osien erottaminen moduuleiksi ja rajapintojen luominen	Cho ja Lee [2004]
		Valmiit erotellut ohjelmistokomponentit	Zhou *et al.* [2010]
		Tasoista muodostuva eroteltu modulaarinen rakenne	Zhou *et al.* [2010]
		Integraatiorajapintojen ja kommunikaation kuvaaminen ja määrittäminen	Zhou *et al.* [2010]

		Valmiit sovelluskehykset SOA-arkkitehtuurilla toteutettuina	Fang ja Sing [2009]
Asiakas-palvelin-arkkitehtuurin käyttö	Skaalautuvuusongelmat Integraatio muihin järjestelmiin vaikeaa Joustavuuden puuttuminen	SOA-pohjainen arkkitehtuuri	Fang ja Sing [2009]
		Hajautettu P2P-pohjainen arkkitehtuuri	Cho ja Lee [2004]
Bisneslogiikan sitominen toteutukseen	Järjestelmien kertakäyttöisyys Vaikeudet suunnitelman ja koodin yhteensovittamisessa Sisällön siirto toisiin järjestelmiin hankalaa	Bisnestapausten erottaminen omiksi moduuleikseen	Duan *et al.* [2006]
		Bisneslogiikan erottaminen omiksi palveluikseen	Fang ja Sing [2009]
Käyttäjänäkökulman tai kustomoinnin ohittaminen	Järjestelmien kertakäyttöisyys Toiminnallisuuksien muuttaminen vaatii uudelleenohjelmointia Käyttäjien mahdollisuudet vaikuttaa sisältöön vähäiset	Käyttötapauksien kuvaaminen	Cong *et al.* [2010]
		Toteutuksien erottaminen omiksi kokonaisuuksikseen	Zhou *et al.* [2010]
		Valmiiden komponenttien käyttö ja ajonaikainen lisääminen/poistaminen	Zhou *et al.* [2010]
Elinkaariajattelun puute	Bisneslogiikkaa ei voida muuttaa muuttamatta järjestelmää Järjestelmän päivitys, integraatio tai siirto uudelle alustalle on vaikeaa tai mahdotonta Päivitysprojektit työläitä ja kalliita	Elinkaariajattelun mukaantuominen arkkitehtuurimallin määrittelyyn	Cong *et al.* [2010]
Arkkitehtuurimallin puute	Ratkaisun miettiminen työlästä Toteutuksissa voi jäädä huomioimatta oleellisia asioita Riskit kokonaisuuden onnistumisessa	SCORM-pohjainen toteutus	Cho ja Lee [2004]
		LTSA-pohjainen toteutus	Cong *et al.* [2010]

Kuva 25. Ongelmat vs. ratkaisut.

On huomionarvoista, että muutamassa tapauksessa eri ongelmien ratkaisuina voi olla sama asia, jolloin näillä ratkaisuilla voidaan nähdä olevan suurempi merkitys kokonaisuuden kannalta kuin muilla. Erimerkkinä tästä on mm. Duanin *et al.* [2006] määrittelemä bisneslogiikan erottaminen toteutuksesta, mikä tuo mukanaan sekä paremman järjestelmän elinkaaren että helpomman järjestelmän integraation ja päivittämisen.

6.4. Jatkotutkimus

Finkin [2005] mukaan jatkotutkimusaiheiden listaamisen tarkoituksena on kuvata artikkeleista löytyneitä asioita, joiden sisältö ei tavalla tai toisella mahtunut tai sopinut tutkimuksen piiriin, mutta jotka kirjoittaja näkee mahdollisina selkeinä erotettavina kokonaisuuksina, joita tulisi tutkia erikseen.

Läpi artikkeleiden ja löydösten oli havaittavissa, että tietyt asiat oli käsitelty artikkeleissa hyvin lyhyesti, tai niitä ei ollut avattu ollenkaan. Näitä olivat esimerkiksi:

- Modulaarisuuden ja integraation tutkiminen ja analysointi e-oppimisratkaisuissa
- Selkeiden ohjelmointistandardien ja ohjelmointikielten kartoitus e-oppimisratkaisuissa

Osa kirjoittajista ei perustellut valintojaan kovinkaan analyyttisesti, joten tältä osin olisi hyvä tehdä lisää tutkimusta ainakin seuraavista aiheista:

- Erilaisten (kilpailevien) e-oppimisarkkitehtuureiden selkeä vertailu keskenään
- (Laajempi) tutkimus käytössä olevista e-oppimisjärjestelmistä ja niiden toteutusratkaisuista

Kaiken kaikkiaan arkkitehtuurinäkökulmasta tutkittuna e-oppimisjärjestelmissä riittäisi selkeästi vielä paljon käsiteltävää sekä bisnes- että toteutusnäkökulmasta niin arkkitehtuuri- kuin standardointimielessä.

Ottaen huomioon, että monet artikkelit toivat esille samoja teemoja, kuten modulaarisuus ja sisällön uudelleenkäytettävyys, voisi olla tarpeen tutkia ja määritellä best practices -käytännöt e-oppimisjärjestelmien toteutuksiin. Ainakin voisi olla tarvetta määritellä best practices -käytännöt nojaten jo olemassa oleviin IEEE:n ja muiden organisaatioiden standardeihin ja malleihin. Myös vertaileva tutkimus käyttäjänäkökulmasta olemassa olevien järjestelmien ja valintojen suhteen voisi tuoda kiinnostavaa tietoa siitä, miten suunnitteluvaiheessa tehdyt ratkaisut ovat tuottaneet tulosta käyttökokemuksen osalta.

Yhtenä mahdollisena tutkimusaiheena voisi olla myös historiatutkimus ja sen avulla muiden tieteenalojen kehityksen vertailu e-oppimisen kehittymiseen tieteenalana, ja kyseisen tutkimuksen peilaaminen muihin vastaaviin teknologioihin ja niiden kehittymiseen. Tällaisen tutkimuksen tuloksena voitaisiin saada arvokasta tietoa siitä, kuinka kypsä ala on, ja mitä vaikutuksia tällä voisi olla e-oppimisjärjestelmiin ja niiden toteutuksiin.

Lisäksi tutkittavana voisi olla myös terminologian käyttö ja se, miten ihmiset kokevat ja määrittelevät e-oppimisjärjestelmät. Varsinkin yritysten käytössä on hyvin kirjava joukko erilaisia järjestelmiä, jotka mielletään e-oppimisjärjestelmiksi, mutta jotka eivät sellaisia suoranaisesti ole. Tälläisissä tapauksissa terminologiaa olisi hyvä tarkentaa ja tarkistaa, jotta varsinkin käyttäjillä olisi selkeämpi kuva siitä millaisessa ympäristössä he toimivat.

Kaiken kaikkiaan koostaville ja vertaileville tutkimuksille aiheen parista olisi huomattavasti enemmän tarvetta kuin nykyisellään toisistaan irtonaisina toteutetuille näkökulmille erilaisiin teknologia- ja arkkitehtuurivalintoihin.

7. Viiteluettelo

[Ambler, 2004] Scott W. Ambler. *The Object Primer: Agile Model-Driven Development with UML 2.0*. Third edition. Cambridge University Press, 2004.

[Almaoui ja Plataniotis, 2005] Mazen Almanoui ja Konstantinos N. Plataniotis. Scalable e-Learning Multimedia Adaptation Architecture. *Image Analysis and Recognition.* Springer Berlin Heidelberg, 2005, 191-198.

[ANSI/IEEE, 2000] ANSI/IEEE. Std 1471-2000, IEEE Recommended Practice for Architectural Description for Software-Intensive Systems. 2000. Saatavilla: *https://standards.ieee.org/findstds/standard/1471-2000.html*.

[Arasteh *et al.*, 2014] Bahman Arasteh, Sajjad Piranesh ja Abdolnasser Zakerib. Highly available and dependable e-learning services using grid system. *Procedia - Social and Behavioral Sciences*, **Volume 143** (2014), 471-476.

[Bizonova *et al.*, 2007] Zuzana Bizonova, Daniel Ranc ja Matilda Drozdova. Model driven e-learning platform integration. *2nd PROLEARN Doctoral Consortium in Technology Enhanced Learning Crete*, (2007), 8-15.

[Chaoa *et al.*, 2015] Kuo-Ming Chaoa, Anne E. Jamesa, Antonios G. Nanosa, Jen-Hsiang Chena, Sergiu-Dan Stanb, Ionut Munteanb, Giorgio Figliolinic, Pierluigi Reac, Chedli B. Bouzgarroud, Pavel Vitliemove, Joshua Cooperf ja Jürgen van Capelleg. *Cloud e-learning for mechatronics: CLEM. Future Generation Computer Systems*, **Volume 48** (2015), 45-59.

[Cho ja Lee, 2004] Su-Jin Cho ja Seongsoo Lee. A module-based software framework for e-learning over internet environment. *Intelligent Tutoring Systems*. Springer Berlin Heidelberg, 2004, 803-805.

[Costagiola *et al.*, 2007] Gennaro Costagliola, Giovanni Casella, Filomena Ferrucci, Giuseppe Polese ja Giuseppe Scanniello. A SCORM thin client architecture for e-learning systems nased on web services. *International Journal of Distance Education Technologies*, **5** (2007), 19-36.

[Cong *et al.*, 2010] Xiao Cong, Hongmei Zhang, Dongdai Zhou, Peng Lu ja Ling Qin. A model-driven architecture approach for developing e-learning platform. *Entertainment for Education: Digital Techniques and Systems*, (August, 2010), 111-122.

[Duan *et al.*, 2006] Bing Duan, Hosseini M. Habib, Voon L. Keck, ja Robert K. Gay. An architecture for online laboratory e-learning system. *Journal of Distance Education Technologies*, **4(2)** (2006), 87-101.

[Fang ja Sing, 2009] Chua F. Fang ja Lee C. Sing. Collaborative learning using service-oriented architecture: A framework design. *Knowledge-Based Systems*, **Volume 22, Issue 4** (May, 2009), 271–274.

[Fink, 2005] Arlene G. Fink. *Conducting Research Literature Reviews. From the Internet to Paper*. Second Edition. Sage Publications Inc., 2005.

[Gaeta, 2007] Angelo Gaeta, Matteo Gaeta ja Pierluigi Ritrovato. A grid based software architecture for delivery of adaptive and personalised learning experiences. *Personal and Ubiquitous Computing*, **Volume 13, Issue 3** (2007), 207-217.

[Gierlowski ja Nowicki, 2007] Krzysztof Gierlowski ja Krzysztof Nowicki. A novel architecture for e-learning knowledge assessment systems. *Advances in Web Based Learning – ICWL*. Springer Berlin Heidelberg, 2007, 276-278.

[Hart, 1999] Chris Hart. *Doing a Literature Review: Releasing the Social Science Research Imagination*. First edition. Sage Publications Ltd., 1999.

[Okolin ja Schabram, 2010] Chitu Okoli and Kira Schabram. A guide to conducting a systematic literature review of information systems research. *Sprouts: Working Papers on Information Systems*, **10(26)** (2010).

[OMG, 2015] Object Management Group. Universal Markup Language. 2015. Saatavilla: *http://www.uml.org/*.

[Kalliala, 2002] Eija Kalliala. *Verkko-opettamisen käsikirja*. Finn Lectura, 2002.

[Kallio, 2006] Tomi J. Kallio. Laadullinen review-tutkimus metodina ja yhteiskuntatieteellisenä lähestymista-pana. *Hallinnon tutkimus*, **2** (2006), 18-28.

[Malik, 2006] Nick Malik. Definition of an architectural model. 2006. Saatavilla: *http://blogs.msdn.com/b/nickmalik/archive/2006/09/17/definition-of-an-architectural-model.aspx*.

[Rouse, 2005] Margaret Rouse. Web-based training (e-learning). 2005. Saatavilla: *http://searchsoa.techtarget.com/definition/Web-based-training*.

[Rouse, 2005] Margaret Rouse. J2EE (Java 2 Platform, Enterprise Edition). 2005. Saatavilla: *http://searchsoa.techtarget.com/definition/J2EE*.

[Rouse, 2014] Margaret Rouse. SOAP (Simple Object Access Protocol). 2014. Saatavilla: *http://searchsoa.techtarget.com/definition/SOAP*.

[Tampereen Yliopisto, 2015] Tampereen Yliopisto. Tiedonlähteet. 2015. Saatavilla: *http://www.uta.fi/kirjasto/oppaat/tiedonhankinnanperusteet/sis/tiedonlahteet/index.html*.

[Turani ja Calvo, 2006] Aiman Turani, Rafael A. Calvo. Beehive: a software application for synchronous collaborative learning. *Campus-Wide Information Systems*, **23(3)** (2006), 196-209.

[W3C, 2015] World Wide Web Consortium. Extensible Markup Language (XML). 2015. Saatavilla: *http://www.w3.org/XML/*.

[Wang *et al.*, 2014] Wang Shunye, Liu Dayong ja Zhang Zijuan. E-learning system architecture based on private cloud for university. *Journal of Chemical and Pharmaceutical Research*, **6(5)** (2014), 492-498.

[Wikipedia, 2015] Wikipedia. Learning Object Metadata. 2015. Saatavilla: *http://en.wikipedia.org/wiki/Learning_object_metadata*.

[Yau *et al.*, 2005] H. K. Yau, E.W.T. Ngai ja T.C.E. Cheng. Conceptual framework and architecture for agent-oriented knowledge management supported e-learning systems. *International Journal of Distance Education Technologies*, **Volume 3, Issue 2** (2005).

[Zhou *et al.*, 2008] Dongdai Zhou, Zhuo Zhang, Shaochun Zhong ja Pan Lie. The design of software architecture for e-learning platforms. *Technologies for E-learning and Digital Entertainment*, (June, 2008), 32-40.